Nemzeti
Közszolgálati Egyetem
Vezető- és Továbbképzési Intézet

BALOGH ZSOLT GYÖRGY – BESZÉDES ÁRPÁD –
BÉRCZES ATTILA – GERGELY TAMÁS –
LEITOLD FERENC – PETHŐ ATTILA –
SZŐKE GERGELY LÁSZLÓ

# Éves továbbképzés az elektronikus információs rendszer biztonságáért felelős személy számára

Budapest, 2014

A tananyag az ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel című projekt keretében készült el.



**Felelős kiadó:**
Patyi András
rektor




ISBN 978-615-5491-66-5

## Tartalom

Nemzeti
Közszolgálati Egyetem
Vezető- és Továbbképzési Intézet

BÉRCZES ATTILA – PETHŐ ATTILA

# Kriptográfia

Budapest, 2014

A tananyag az ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel című projekt keretében készült el.

**Szerző:**
© Bérczes Attila – Pethő Attila 2014

**Kiadja:**
© NKE, 2014

**Felelős kiadó:**
Patyi András
rektor





# Tartalom

# Bevezetés

A kriptográfia a bizalmas információváltás algoritmusaival foglalkozó tudomány. Nélkülözhetetlen szerepet játszik az adatok biztonságos tárolásában és továbbításában. A titkosítás informálisan értelmes üzenetek értelmetlen betűsorozatokba transzformálását jelenti. Ha például egy puha agyaglapra üzenetet karcolunk, majd az agyaglapot kiégetjük, akkor az üzenet jól olvasható. Kalapáccsal apró darabokra törve a lapot és a szilánkokat jól összekeverve azonban az üzenet csak igen fáradságos munkával nyerhető vissza, ha egyáltalán visszanyerhető. A kriptográfiai algoritmusok a kalapácsnál sokkal alaposabban kuszálják össze az üzenetek struktúráját és teszik értelmetlenné azokat.

A gondolkodó olvasó ellenveti, hogy az agyagtábla szétverése irreverzibilis beavatkozás, ha jól végezzük a dolgunkat, akkor senki sem tudja visszaállítani az eredeti üzenetet. Ennek pedig semmi értelme, hiszen a bizalmas üzenetet a címzettnek el kell olvasnia. Az ellenvetés jogos is meg nem is. Jogos, mert az alkalmazások jelentős részében valóban vissza akarjuk nyerni a titkosított üzenetből az eredetit. Vannak azonban olyan – hétköznapi – alkalmazások, pl. az azonosítás, amikor kifejezetten káros a titok visszanyerhetősége. Az agyaglapos példánkban nem is a megfejthetőség a lényeg, hanem az, hogy a titkosítás az értelmes információ összekuszálását jelenti.

A titkos információcserét évezredek óta alkalmaznak az államok és a hadseregek vezetői, forradalmárok és kémek, de szerelmesek is használták, nehogy idő előtt kitudódjon a titkuk. Nagyon sokféle technikát dolgoztak ki titkosításra, beleértve a mechanikus és elektromechanikus eszközöket is. A téma iránt érdeklődő olvasó figyelmébe ajánljuk Simon Singh kitűnő könyvét[1].

A múlt század második felében tömeges igény jelent meg először a gazdaság, majd a közigazgatás szereplői részéről is a bizalmas adattovábbításra. Katonai és államtitkok helyett pénzügyi tranzakciók, vállalati titkok, érzékeny személyi adatok, stb. igényelték a bizalmas adattovábbítást. A tömeges adatcsere technikai lehetőségét a telefon- majd a számítógép hálózatok teremtették meg. Az 1970-es években kezdték el a titkosító algoritmusok szisztematikus kutatását. Az IBM nyilvánosságra hozta a DES-t (ld. 2.2.1. fejezet), amelyik néhány éven belül szimmetrikus titkosítási szabvány lett és az is maradt a legutóbbi időkig. A DES a korábbi titkosító algoritmusok közvetlen leszármazottja. Lényegében többféle, régóta ismert és használt titkosítási módszer összekapcsolása. Újdonságát az jelentette, hogy elektronikai eszközökre optimalizálták.

A titkosítási algoritmusokra jelentkező tömeges igényt kizárólag szimmetrikus algoritmusokkal nem lehet kielégíteni. A titkosításhoz és megfejtéshez szükséges kulcsot ugyanis mindkét kommunikálni szándékozó félnek ismerni kell. Vagy előre meg kell tehát állapodniuk a kulcsban vagy valamilyen védett csatornán kell eljuttatni hozzájuk a kulcsot. A szükséges kulcsok száma a partnerek számának négyzetével arányos, ezért az első út nem járható. Lehallgathatatlan

[1] Simon Singh, Kódkönyv, A rejtjelezés és a rejtjelfejtés története, Park Kiadó, 2002.

csatornákat használnak, de ezek meglehetősen költségesek. Elvileg is új megoldás után kellett tehát nézni. Whitfield Diffie és Martin Hellman 1976-ban vetette fel a nyilvános kulcsú vagy aszimmetrikus titkosítás forradalmian új ötletét. Az első, máig is biztonságosnak tekinthető aszimmetrikus titkosítási algoritmust Ron Rivest, Adi Shamir és Leonhard Adleman publikálta 1978-ban, amely RSA néven vonult be a kriptográfia történetébe. A kriptográfia modern története ezzel a két dolgozattal kezdődött és mára terebélyes elméletté vált. Fontossága az internet terebélyesedésével egyenes arányban növekedett. Az internet ugyanis egy óriási nyilvános hálózat, amelyet a múlt század utolsó évtizedében a gazdasági élet szereplői is felfedeztek és kihasználták a benne levő lehetőségeket. A kriptográfia eredményei nem látványosak, de nélkülük a világháló nem fejlődhetett volna viharosan. A jegyzetben számos példát mutatunk be a kriptográfia hétköznapi alkalmazásaira és a sort még bőségesen folytathatnánk. A legmarkánsabb példa a digitális aláírás, amelyet Diffie és Hellman a fentebb idézett cikkben tárgyalt először, és amely a XXI. században minden ország jogrendjének részévé vált.

Jegyzetünkben csak rövid áttekintést tudunk adni a kriptográfia kiterjedt elméletéről és gyakorlatáról. A legfontosabb alkalmazásokra: azonosítás, kulcscsere, digitális aláírás és titkosítás koncentrálunk és ezekre is csak az alapvető eljárásokat tárgyaljuk. A kriptográfiai algoritmusok komoly matematikai elméleteken alapulnak. A megfogalmazásukhoz nélkülözhetetlen matematikai ismereteket az 1. fejezetben gyűjtöttük össze. A 2. fejezetben a szimmetrikus titkosítással, a 3. fejezetben pedig az azonosítással foglalkozunk. A 4. fejezet témája az aszimmetrikus titkosító eljárások. Itt foglalkozunk a kulcscserével és a digitális aláírás elvi modelljével is. Az 5. fejezet a digitális aláírás működtetéséhez nélkülözhetetlen nyílt kulcs infrastruktúra elemeit mutatja be. Jegyzetünk Fogalomtárral fejeződik be.

# 1 Matematikai alapismeretek

A modern titkosító algoritmusok megfogalmazásához feltétlenül szükséges matematikai ismereteket foglaltuk össze ebben a fejezetben. Ezek némelyike része a középiskolás tananyagnak. A jelölések és fogalmak egységes használata miatt szükségesnek tartottuk a feltételezhetően ismert tudásanyagot is áttekinteni.

## 1.1 Természetes számok helyiértékes ábrázolása

A matematika történetének egyik jelentős fordulópontja volt az arab számok bevezetése, ugyanis ez a számok olyan ábrázolását jelentette, amelyben a szám „kinézete" és értéke szoros kapcsolatban volt egymással. Ez egyrészt lehetővé tette, hogy a 10 számjegy segítségével akármekkora számokat megjelenítsenek (ráadásul „helytakarékosan"), másrészt az alapműveletek elvégzését is nagyon megkönnyítette. Ebben a rendszerben az volt forradalmian új, hogy minden természetes számot a 0,1,2,3,4,5,6,7,8,9 számjegyek megfelelő módon történő egymásután írása segítségével ábrázolja, és a számjegyek értéke attól függ, hogy hányadik helyen állnak:

- a legutolsó számjegy annyit ér, mint amennyi az értéke
- az utolsó előtti annyit ér, mint értékének a tízszerese
- hátulról a harmadik százszor annyit ér, mint az értéke
- ……..
- hátulról a k-adik $10^{k-1}$-szer annyit ér, mint az értéke.

Ez az oka annak, hogy ezt az ábrázolási módot helyiértékes ábrázolásnak is hívják.

Például a 9375 szám értéke $5 + 7 \cdot 10^1 + 3 \cdot 10^2 + 5 \cdot 10^3$.

Ezt általánosabban úgy lehet megfogalmazni, hogy minden $n$ természetes szám egyértelműen felírható az alábbi alakban:

$$n = \sum_{i=0}^{k} a_i \cdot 10^i,$$

ahol az $a_i$ számok a $\{0,1,2,3,4,5,6,7,8,9\}$ halmazból kerülnek ki. Ekkor az $n$ szám jelölésére az $\overline{a_k a_{k-1} \dots a_1 a_0}$ számsorozatot használjuk. Mivel a fenti képletben a 10 hatványai szerepelnek, az $\overline{a_k a_{k-1} \dots a_1 a_0}$ alakot az $n$ szám 10-es számrendszerbeli alakjának is nevezzük. A felülvonást természetesen csak akkor használjuk, ha az általánosság kedvéért a számjegyek helyett betűket írunk, mert így nem téveszthető össze a szám 10-es számrendszerbeli alakja számjegyek $a_k a_{k-1} \dots a_1 a_0$ szorzatával.

Szemléletesen látható, és természetesen matematikailag is igazolható, hogy a 10-es szám helyett a számrendszerünk alapjául bármelyik másik $g \geq 2$ számot választhatjuk, de ekkor a számjegyek a $\{0,1,\dots,g-1\}$ halmazból kerülhetnek ki. Igaz továbbá, hogy minden $n$ természetes szám egyértelműen felírható az alábbi alakban:

$$n = \sum_{i=0}^{k} a_i \cdot 10^i,$$

ahol az $a_i$ számok a $\{0,1,\dots,g-1\}$ halmazból valók.

Amennyiben nem egyértelmű, hogy egy számnak melyik számrendszerbeli alakját írtuk fel, akkor a szám mögé írjuk indexben a számrendszer alapját. Például a hetes számrendszerbeli $213_7$ szám a $2 \cdot 7^2 + 2 \cdot 7 + 3 = 115_{10}$ természetes számnak felel meg, ahol a 115 mögé nem feltétlenül szükséges az indexben a 10-et leírni, ha úgy érezzük, hogy a szövegkörnyezetből világos, hogy a szám 10-es számrendszerbeli alakjáról van szó.

Ha a továbbiakban helyiértékes ábrázolással megadunk egy természetes számot és mást nem mondunk, akkor az 10-es számrendszerben lévő számnak tekintendő.

*Tesztkérdések:*

T1. Írjuk fel a $13$ természetes számot a kettes számrendszerben.
T2. Melyik tízes számrendszerbeli szám kettes számrendszerbeli alakja a $10010111_2$?

## 1.2 Számelméleti alapfogalmak

***Definíció.*** Azt mondjuk, hogy a $d$ egész szám osztója az $a$ egész számnak, ha létezik olyan $q$ egész szám melyre $a = bq$. Ekkor a $d \mid a$ jelölést használjuk.

***Tétel.*** **(A maradékos osztás tétele)** Tetszőleges $a$ és $b \neq 0$ egész számokhoz egyértelműen léteznek olyan $q$ és $r$ egész számok, melyekre

$$a = qb + r \qquad \text{és} \qquad 0 \leq r < |b|.$$

***Definíció.*** Azt mondjuk, hogy a $d$ egész szám az $a$ és $b$ egész számok legnagyobb közös osztója, ha teljesíti az alábbi két feltételt

1. $d \mid a$ és $d \mid b$, valamint
2. ha valamely $d'$ egész számra $d' \mid a$ és $d' \mid b$, akkor $d' \mid d$.

Ekkor a $d = (a, b)$ jelölést használjuk.

***Definíció.*** Azt mondjuk, hogy az $a$ és $b$ egész számok relatív prímek egymáshoz, ha legnagyobb közös osztójuk $1$.

***Tétel.*** **(Az Euklideszi algoritmus)** Legyen $a$ és $b$ két tetszőleges egész szám. Ekkor létezik $a$ és $b$ legnagyobb közös osztója, és ez az alábbi algoritmussal kiszámítható:

*Végezzünk sorozatos maradékos osztásokat, úgy hogy kezdetben* $a$ *az osztandó és* $b$ *az osztó, majd minden új lépésben az osztandó az előző osztó, az osztó pedig az előző maradék legyen. Ezt addig folytassuk, míg nulla maradékot kapunk. Ekkor a legutolsó nem-nulla maradék adja meg*$a$ *és* $b$ *legnagyobb közös osztóját,* azaz ha

$$\begin{array}{cc} a = bq_1 + r_1 & 0 < r_1 < |b| \\ b = r_1 q_2 + r_2 & 0 < r_2 < r_1 \\ r_1 = r_2 q_3 + r_3 & 0 < r_3 < r_2 \\ \dots\ \dots & \dots\ \dots \\ r_{n-2} = r_{n-1} q_n + r_n & 0 < r_n < r_{n-1} \\ r_{n-1} = r_n q_{n+1} & r_{n+1} = 0 \end{array}$$

Ekkor $(a, b) = r_n$.

***Definíció.*** Legyen $p \neq 0, \pm 1$ egy egész szám. Ha $p$ összes osztója $\pm 1$ és $\pm p$, akkor a $p$ számot prímszámnak nevezzük. Ha egy $a \neq 0, \pm 1$ szám nem prímszám, akkor összetett számnak nevezzük.

Az első néhány prímszám: $2,3,5,7,11,13, \dots$ Két évezrede Eukleidész bizonyította be, hogy végtelen sok prímszám van.

***Tétel.*** **(Az egyértelmű prímfaktorizáció tétele)** Minden $a \neq 0, \pm 1$ egész szám a tényezők sorrendjétől és $\pm 1$ szorzóktól eltekintve egyértelműen felírható prímszámok szorzataként. Az

$$a = e\, p_1^{u_1} p_2^{u_2} \dots p_s^{u_s}$$

alakot - ahol $e = \pm 1, p_1, \dots, p_s$ különböző prímek és $u_1, \dots, u_s$ pozitív egészek -, az $a$ szám kanonikus alakjának nevezzük.

***Definíció.*** Minden $x$ valós szám esetén jelölje $\pi(x)$ az $x$-nél nem nagyobb pozitív prímek számát. Az így definiált $\pi$: $\mathbb{R} \to \mathbb{R}$függvényt a prímek száma függvénynek nevezzük.

***Tétel.*** **(Csebisev tétele)** Léteznek olyan $c_1$ és $c_2$ konstansok, melyekre

$$c_1 \frac{x}{\log x} < \pi(x) < c_2 \frac{x}{\log x}.$$

*Tesztkérdések:*

T3. Mennyi a $132$ természetes szám $16$-tal való osztási maradéka?

T4. Mire alkalmazható az Euklideszi algoritmus?

a) Egy szám faktorizálására.
b) Egy szorzás gyorsabb elvégzésére.
c) Két szám legnagyobb közös osztójának a meghatározására.
d) Nagy számok összeadására.

T5. Az alábbi számok közül melyek relatív prímek a 42-höz?

a) 16
b) 33
c) 91
d) 55

## 1.3 Kongruenciák és tulajdonságaik

***Definíció.*** Legyenek $a$ és $b$ egész számok, és $m$ egy pozitív egész szám. Ekkor azt mondjuk, hogy $a$ kongruens $b$-vel modulo $m$, ha $m$ osztja az $a-b$ számot, azaz $m|a-b$. Ekkor az alábbi jelölést használjuk

$$a \equiv b \pmod{m}.$$

Az $m$ számot ekkor a kongruencia modulusának nevezzük.

***Példa.*** $7 \equiv 29 \pmod{11}$ mivel $29 - 7 = 2 \cdot 11$.

$3 \not\equiv 8 \pmod{2}$ mivel $8 - 3$ nem osztható 2-vel.

A kongruenciák alábbi tulajdonságai lényegében az oszthatóság tulajdonságainak egyszerű következményei.

***Tétel.*** **(A kongruencia tulajdonságai)** Legyenek $a, b, c$ egész számok, $m$ pedig egy pozitív egész szám. Ekkor

1. $a \equiv b \pmod{m}$ akkor és csakis akkor, ha $a$ és b ugyanazt az osztási maradékot adja $m$-mel való osztás során,
2. $a \equiv a \pmod{m}$ minden $a$ egész szám és minden $m$ modulus esetén,
3. ha $a \equiv b \pmod{m}$ akkor $b \equiv a \pmod{m}$,
4. ha $a \equiv b \pmod{m}$ és $b \equiv c \pmod{m}$ akkor $a \equiv c \pmod{m}$,
5. ha $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m}$ akkor $a + c \equiv b + d \pmod{m}$,
6. ha $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m}$ akkor $a - c \equiv b - d \pmod{m}$,
7. ha $a \equiv b \pmod{m}$ és $c \equiv d \pmod{m}$ akkor $ac \equiv bd \pmod{m}$,
8. ha $a \equiv b \pmod{m}$ akkor $a + c \equiv b + c \pmod{m}$,
9. ha $a \equiv b \pmod{m}$ akkor $a - c \equiv b - c \pmod{m}$,

10. ha $a \equiv b \pmod{m}$ akkor $ac \equiv bc \pmod{m}$,
11. ha $a \equiv b \pmod{m}$ akkor minden $n$ természetes szám esetén $a^n \equiv b^n \pmod{m}$,
12. ha $f$ egy egész együtthatós polinom és $a \equiv b \pmod{m}$ akkor $f(a) \equiv f(b) \pmod{m}$.

Az alábbi állítás azt mutatja be, hogy milyen szabályok szerint lehet egy kongruencia mindkét oldalát elosztani egy számmal:

***Tétel.*** Legyenek $a, b, c$ egész számok, és $m$ pozitív egész szám. Jelölje $d$ az $m$ és $c$ számok legnagyobb közös osztóját. Ekkor

$$\text{ha } ac \equiv bc \pmod{m} \text{ akkor } a \equiv b \left(\mathrm{mod}\,\frac{m}{d}\right).$$

***Definíció.*** Legyen $a$ és $b$ két nem-nulla egész szám. Azt mondjuk, hogy $a$ és $b$ relatív prímek egymáshoz, ha legnagyobb közös osztójuk 1, azaz ha $(a, b) = 1$.

***Definíció.*** Legyen $n$ egy pozitív egész szám. Jelölje $\varphi(n)$ azon 1 és $n$ közé eső egész számok számát, melyek relatív prímek n-hez. Az így definiált $\varphi: \mathbb{N} \to \mathbb{N}$ függvényt az Euler-féle $\varphi$-függvénynek nevezzük.

***Tétel.*** Legyen $n$ egy természetes szám, melynek prímfelbontása $n = p_1^{u_1} \dots p_s^{u_s}$. Ekkor

$$\varphi(n) = \prod_{i=1}^{s}\left(p_i^{u_i} - p_i^{u_i-1}\right) = \prod_{i=1}^{s} p_i^{u_i-1}(p_i - 1) = n\prod_{i=1}^{s}\left(1 - \frac{1}{p_i}\right).$$

***Példa.*** $\varphi(6) = 2$, $\varphi(12) = 4$, $\varphi(144) = 144\left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{3}\right) = 48$,
ha $p$ prímszám, akkor $\varphi(p) = p - 1$.

***Euler-Fermat tétel.*** Legyen $a$ egy egész szám és $m$ egy pozitív egész szám. Ha $(a, m) = 1$ akkor

$$a^{\varphi(m)} \equiv 1 \pmod{m}.$$

***Tétel.*** Legyenek $a$ és $b$ egész számok, $m$ pedig egy pozitív egész szám. Tekintsük az

$$ax \equiv b \pmod{m}$$

lineáris kongruenciát. A fenti kongruencia-egyenletnek akkor és csakis akkor van megoldása, ha $(a, m) \mid b$, azaz, ha $a$ és $m$ legnagyobb közös osztója osztja a $b$ számot. Ekkor a kongruenciának pontosan $(a, m)$ darab modulo $m$ páronként inkongruens megoldása van.

***Példa.*** Oldjuk meg az alábbi lineáris kongruenciát:

$$5x \equiv 17 \pmod{19}$$

Megoldás:Mivel $(5,19) = 1 \mid 17$, így a kongruenciának egyértelmű megoldása van modulo 19. A megoldáshoz keresünk egy olyan $k$ egész számot, melyre $k \cdot 19 + 17$ osztható 5-tel. Például $k = 2$ megfelelő választás. Tehát

$$5x \equiv 17 \pmod{17}$$
$$5x \equiv 17 + 2 \cdot 19 \pmod{17}$$

$$5x \equiv 55 \pmod{17}$$
$$x \equiv 11 \pmod{17}.$$

***Definíció.*** Legyen $m$ egy pozitív egész szám. Azt mondjuk, hogy egy $a$ egész számnak létezik inverze modulo $m$ (vagy más szóval $a$ invertálható modulo $m$), ha létezik olyan $b$ egész szám melyre

$$ab \equiv 1 \pmod{m}.$$

teljesül.

***Példa.*** Létezik-e 7-nek inverze modulo 15? A válasz igen, mivel $7 \cdot 13 \equiv 1 \pmod{15}$. Létezik-e 2-nek inverzemodulo 12? A válasz nyilván nem, mivel 2 bármely többszöröse páros, ami modulo 12 csak páros számmal lehet kongruens.

***Tétel.*** Legyen $a$ tetszőleges egész szám és $m$ egy pozitív egész szám.Az $a$ számnak akkor és csakis akkor létezik inverze modulo $m$, ha (a,m)=1.

***Jelölés.*** Kriptográfiai, illetve informatikai alkalmazások során az

$$a \bmod m$$

kifejezés alatt azt a legkisebb nem-negatív számot értjük, amelyik kongruens $a$-val modulo $m$. Hasonlóan az

$$a^{-1} \bmod m$$

azt a legkisebb nem-negatív számot jelöli, mely inverze $a$-nak modulo $m$.

*Tesztkérdések:*

T6. Az alábbi számok közül melyek kongruensek 13-mal modulo 16?
   a) 45
   b) 31
   c) 61
   d) 23

T7. Számítsa ki $\varphi(120)$ értékét!

T8. Az alábbi számok közül melyeknek létezik inverze modulo 60?
   a) 55
   b) 8
   c) 54
   d) 11

T9. Mennyi az 5 inverze modulo 16?
   a) 5
   b) 13
   c) 3
   d) 8

## 1.4 Permutációk

*Definíció.*Legyen $A$ egy véges halmaz. Az $A$ egy permutációja alatt egy $\sigma: A \to A$ bijektív függvényt értünk.

*Megjegyzés.* Az $A$ véges halmaz egy permutációja lényegében $A$ elemeinek egy sorba rendezését jelenti. Az $A = \{a_1, \dots, a_n\}$ halmaz permutációit $(b_1, \dots, b_n)$ szám $n$-esekkel jelöljük, ahol a $b_1, \dots, b_n$ elemek éppen az $A$ halmaz különböző elemei.

*Példa.* Legyen $A = \{1,2,3,4\}$. A összes lehetséges permutációi a következők:
$(1,2,3,4), (1,2,4,3), (1,3,2,4), (1,3,4,2), (1,4,2,3), (1,4,3,2), (2,1,3,4), (2,1,4,3),$
$(2,3,1,4), (2,3,4,1), (2,4,1,3), (2,4,3,1), (3,1,2,4), (3,1,4,2), (3,2,1,4), (3,2,4,2),$
$(3,4,1,2), (3,4,2,1), (4,1,2,3), (4,1,3,2), (4,2,1,3), (4,2,3,1), (4,3,1,2), (4,3,2,1).$

*Tétel.* Egy $n$-elemű halmaz összes permutációinak száma $n! = 1 \cdot 2 \cdot \dots \cdot n$ .

*Tesztkérdések:*

T10. Hány különböző permutációja van az $\{1,2,3,4,5\}$ halmaznak?

## 1.5 Betűk konvertálása számokká

Egy üzenet titkosítása során valamilyen titkosító algoritmus segítségével olyan alakúra transzformáljuk az üzenetet, hogy az illetéktelen személyek számára olvashatatlan, pontosabban értelmezhetetlen legyen. A modern kriptorendszerek esetén általában a titkosítás biztonságos voltát matematikai ismereteink garantálják, vagy legalábbis valószínűsítik, így számunkra, kriptográfiai szempontból az a kényelmes, ha a titkosítandó üzenet számokból áll. Természetesen az üzenet eredetileg nem feltétlenül áll számokból. Például, ha egy írott szöveget szeretnénk titkosítani, akkor előbb a szöveget valamilyen számsorozattá kell konvertálnunk, erre kell alkalmaznunk a titkosító algoritmust, majd elküldhetjük az üzenetet, amit a címzett visszafejt, így megkapva újra az eredeti számsort, de ezt még olvasható üzenetté kell konvertálnia.

Az egyes kriptorendszerek bemutatása során általában az üzenetek számokká történő konvertálásáról nem ejtünk szót, ugyanis ez nem része a Kriptográfia tudományának. Ugyanakkor itt megemlítjük mik a legfontosabb szempontok a szövegek számmá történő konvertálása során:

- A betűket (vagy betűcsoportokat) úgy kell számoknak megfeleltetni, hogy a megfeleltetés injektív legyen, azaz különböző betűknek (betűcsoportoknak) különböző számok feleljenek meg.
- A konverzió legyen oda-vissza gyorsan elvégezhető.
- Ennek a konverziónak, és szöveggé történő visszaalakításnak a módja közismert, ez nem javítja a titkosítás biztonságát.

***Példa:*** Tegyük fel, hogy a titkosítandó szövegünk magyar nyelven íródott, és vegyük alapul a 44 betűs magyar ábécét:

A, Á, B, C, Cs, D, Dz, Dzs, E, É, F, G, Gy, H, I, Í, J, K, L, Ly, M, N,
Ny, O, Ó, Ö, Ő, P, Q, R, S, Sz, T, Ty, U, Ú, Ü, Ű, V, W, X, Y, Z, Zs.

Ekkor egy lehetséges megfeleltetés betűk és számok között:

A↔0, Á↔1, B↔2, … , Zs↔43.

Természetesen elképzelhető, hogy a szöveget például betűpáronként szeretnénk számokká alakítani. Ekkor $44^2 = 1936$különböző számra lesz szükségünk.

Amennyiben kettes számrendszerbeli számokat szeretnénk kapni, úgy a betűknek (betűcsoportoknak) megfelelő számokat átírhatjuk kettes számrendszerbeli alakjukra.

A fentiek alapján tehát látható, hogy a Kriptográfia szempontjából teljesen mindegy, hogy a titkosítandó üzenet eredeti formája milyen, hiszen azt számsorozattá tudjuk alakítani, és vissza, tehát nyugodtan kiindulhatunk abból, hogy üzenetünk olyan számsorozat formájában adott, ami megfelel a titkosító algoritmusunk inputjának. Az általunk a jegyzet további részeiben tárgyalt kriptorendszerek esetén (egy-két klasszikus kriptorendszer kivételével) feltesszük, hogy az üzenetek számok formájában vannak megadva, de emlékeztetünk arra, hogy semmilyen értelemben nem jelent megszorítást a titkosítható üzenetek tekintetében.

## 1.6 Megjósolhatatlan véletlen számok

Kriptográfiai algoritmusokban és protokollokban gyakran használnak véletlen számokat. Ez a kifejezés azt jelenti, hogy olyan szám, amelyet a lottóhúzás vagy a kockadobás során kapunk. Bármilyen sokat fejlődött azonban a számítástechnika az elmúlt 60 évben, a számítógépek (még) nem tudnak kockával dobni. Elméleti lehetőség sincs arra, hogy a számítógépek által előállított számok a lottó- vagy a rulettszámokhoz hasonlóan véletlenek legyenek, mert a komputerek csak a beprogramozott lépéseket hajthatják végre, a szakzsargon szerint determinisztikusak.

A különböző szerencsejátékok feljegyzett eredményei igazi véletlen számsorozatokat jelentenek. Az ötöslottó első sorsolása 1957. február 20-án volt hazánkban. Az elmúlt közel 57 évben tehát 3000-szer húzták ki az 5 nyerőszámot, ami 15000 darab 1 és 90 közötti számot jelent. A kriptográfiai algoritmusoknak ettől nagyságrendekkel több véletlen számra van szükségük.

Szerencsére nemcsak a kriptográfiának van szüksége véletlen számokra, hanem a szimulációs- és a játékprogramoknak is. A számítógéppel előállítható álvéletlen sorozatok elmélete a múlt század elejére nyúlik vissza. Nagyon egyszerű, de sok gyakorlati alkalmazásban mégis felhasználható a *lineáris kongruencia generátor*. Ezeknél a, b és m jól megválasztott természetes számok és az álvéletlen sorozat tagjait az$an + b \pmod m$ képlettel számítjuk. Az $a = 13, b = 7, m = 61$ választás mellett a sorozat első néhány tagja: 7, 20, 33, 46, 59, 11, … Ajánljuk az olvasónak, hogy

folytassa a számsorozatot. A gyakorlatban természetesen a,b és m lényegesen nagyobb számok, például a C/C++ programozási nyelvek beépített lineáris kongruencia generátora $a = 22695477$, $b = 1$, $m=2^{32}= 4294967296$ értékeket használja.

Lineáris kongruencia generátorok közvetlenül nem alkalmazhatóak a kriptográfiában, mert három egymást követő tagjukat megfigyelve könnyen ki lehet számolni a,b és m értékét, így az ellenfél előre ismeri a titkosító rendelkezésére álló álvéletlen számok sorozatát. Olyan módszerekre van tehát szükségünk, amelyek elég gyorsan számolhatóak, jó statisztikai tulajdonságokkal rendelkeznek, és nemjelezhetőek előre. Utóbbi azt jelenti, hogy a generátor által kibocsátott számokat akármilyen sokáig is megfigyelve ne lehessen előre jelezni a számsorozat további lefolyását.

A számítógép pillanatnyi fizikai jellemzői, például a rendszeridő, az egér pozíciója, a memória foglaltsága, stb. hasonlóan jó minőségű véletlen számokat eredményeznek, mint a kockadobás, de számosságuk kicsi, így elsősorban álvéletlen sorozatok kezdőértékeként jöhetnek szóba.

Az egyik legszélesebb körben alkalmazott, kriptográfiai szempontból is megfelelő, álvéletlen generátort Lenore Blum, Manuel Blum és Michael Shub[2] publikálta 1986-ban. A konstrukció hasonló a később tárgyalandó RSA titkosító algoritmushoz. Legyenek $p$ és $q$ nagy prímszámok, amelyek 3 maradékot adnak 4-el osztva és $m = pq$. Válasszunk egy olyan $x_0$ számot, amelyik *0-ától* és *1*-től különbözik és sem $p$-vel sem $q$-val nem osztható. Ezek után a sorozat számolási szabálya: $x_{n+1} = x_n^2 \ mod \ m$. Ha például $p = 19$ és $q = 31$, akkor $m = 589$. Az $x_0 = 43$ számra a feltételek teljesülnek, ezek a paraméterek tehát – nagyságrendjüktől eltekintve - megfelelnek egy Blum-Blum-Shub generátornak. Az így előálló sorozat első néhány tagja: 43, 82, 245, 536,… A p és q számok megválasztásával kapcsolatban ugyanazt mondhatjuk el, amit az RSA-val kapcsolatban a 4.1 fejezetben meg fogunk tenni.

Sárközy András akadémikus C. Mauduittal[3] álvéletlen sorozatok minőségének mérésére elméleti szempontból is jól kezelhető mérőszámokat vezetett be. Dolgozatuk kiterjedt elméletté vált, olyan pszeudovéletlen bináris sorozatokat sikerült konstruálni, amelyek elméleti és gyakorlati szempontból is jó tulajdonságúak. A szükséges matematikai apparátus hiánya miatt ebben a jegyzetben nem tudjuk az eljárásokat pontosabban bemutatni.

[2]Blum, Lenore; Blum, Manuel; Shub, Mike, A Simple Unpredictable Pseudo-Random Number Generator, *SIAM Journal on Computing***15** (1986): 364–383.

[3]Mauduit, Christian; Sárközy, András, On finite pseudorandom binary sequences. I. Measure of pseudorandomness, the Legendre symbol.Acta Arith.82 (1997), no. 4, 365–377.

*Tesztkérdések:*

T11. Ellenőrző feladat: Nézze meg a számítógépe C: lemeze kapacitását bájtban. Legyen ez n0. Számítsa ki n1, n2, n3 és n4 értékét a C/C++ lineáris kongruencia generátorával!

T12. A Blum-Blum-Shub generátor modulusa
   a) két nagy prímszám szorzata,
   b) prímszám,
   c) két olyan nagy prímszám szorzata, amelyek 3 maradékot adnak 4-el osztva.

## 1.7 A titkosítás matematikai modellje; folyam- és blokktitkosítók, egyirányú és egyirányú csapóajtó függvények

A titkosítás matematikai modelljének kidolgozása és elemzése nem öncélú játék, hanem fontos gyakorlati követelmény. Amíg a bizalmas üzenettovábbítás csak erősen centralizált szervezetek igénye volt, absztrakt modellre nem volt szükség. Alapvetően megváltozott azonban a helyzet az informatikai majd az infokommunikációs hálózatok megjelenésével és elterjedésével. A titkosító eljárásokat ma számítógépen implementálják és bonyolult protokollokba építik be. A titkosítás matematikai modelljének kidolgozása tette lehetővé, hogy a titkosító eljárások és a köréjük épített protokollok biztonságosságát matematikai eszközökkel is elemezni tudjuk. Így pontos ismereteket szerezhetünk az algoritmusaink biztonságáról. A matematikai állítások általános tulajdonsága, hogy feltételesek, csak jól meghatározott premisszák teljesülése esetén igazak. Még így is sokkal szilárdabb alapot jelentenek a mérnökök számára nagy rendszerek kidolgozása során, mint a csak heurisztikára épített ismeretek.

Az (*U,K,T,E,D*) ötöst *kriptográfiai rendszernek* nevezzük, ha *U, K* és *T* véges halmazok, az *E* az $U \times K$ [4] halmaznak *T*-be, míg *D* a $T \times K$ halmaznak *U*-ba való leképezése. Ezen kívül megköveteljük, hogy mind az *E*, mind a *D* leképezés könnyen számítható legyen. Az *U* elemeit *üzenetblokk*oknak, a *K* elemeit *kulcs*oknak, végül *T* elemeit *titokblokk*oknak nevezzük. Az *E* leképezés a titkosító függvény, a *D* pedig a megfejtő vagy dekódoló függvény.

A gyakorlatban *U* és *K* elemei 64, 128, 1024 vagy 2048 darab 0 és 1-ből álló (bináris) szavak, amelyeket az *E*-t és *D*-t kiszámító algoritmus jellegétől függően hol bináris szónak, hol természetes számok bináris alakjának tekintünk. Például a 8 bitből álló 0100 1101 jelsorozat a $2^6 + 2^3 + 2^2 + 1 = 77$ szám bináris alakja.

Általában feltételezzük, hogy minden $k_E$ titkosító kulcshoz legyen egy $k_D$ megfejtő kulcs, amely azzal a tulajdonsággal rendelkezik, hogy ha egy *u* üzenetblokkot a $k_E$ kulccsal titkosítunk, majd az eredményt a $k_D$ megfejtő kulccsal dekódolunk, akkor visszakapjuk az *u*-t. Ezt matematikai formalizmussal a $D(E(u, k_E), k_D) = u$ azonossággal fejezhetjük ki.

[4] Az *UxK* halmaz (*u,k*) párokból áll, ahol *u* az *U*, míg *k* a *K* elemein fut végig.

Az *E* és a *D* függvények értékeit gyorsan ki kell számítani, ami a gyakorlatban annyit jelent, hogy a felhasználó ne érzékelje, hogy a számítógép valamilyen manipulációt végez az adataival. ***A kódolás titkosságát az biztosítja, hogy az E, a D és az E(u, $k_E$) ismeretében az u értékét vagy a megfejtő kulcsot gyakorlatilag lehetetlen legyen kitalálni vagy kiszámítani.*** Ez a tulajdonság a titkosítási rendszerek fő specifikuma. A matematikai formalizmust emberi nyelvre fordítva annyit jelent, hogy hiába ismerjük a titkosított üzenetet $E(u, k_E)$, valamint a titkosítás és a megfejtés módját, a titkot ne lehessen kitalálni. A megfejtő kulcs birtokában azonban könnyen dekódolni lehessen a titkos üzenetblokkot.

Titkosítást évszázadokig kizárólag a katonaság, a titkosszolgálatok és az államok vezetői használtak. A kommunikálni szándékozó felek személyes találkozás alkalmával beszélték meg a titkosítási eljárást, valamint a titkosító és megfejtő kulcsokat. Az internet elterjedésével olyan üzleti, egészségügyi, igazgatási, közszolgálati és egyéb alkalmazások jelentek meg, amelyek csak bizalmas információcserével oldhatóak meg. A személyes adatok, választási részeredmények, adóbevallás, versenypályázatok, banki tranzakciók adatai és más fontos adatok továbbítása csak titkosítva történhet. Ilyenkor a kommunikálni szándékozó szereplők nagy száma lehetetlenné teszi egyedi titkosító eljárások használatát. ***A polgári alkalmazásokban tehát a titkosító és megfejtő függvények szabványosak***. Természetes elvárás, hogy a hazánkban készített lámpába Németországban is lehessen izzót vásárolni. Ugyanilyen elvárás, hogy a hazánkban, szabványos eljárással titkosított üzenetet a dekódoló kulccsal Németországban is meg lehessen fejteni.

Modern kriptorendszerekben tehát az üzenetek titkossága csakis a titkosító és a megfejtő függvény, valamint a titkosító és a megfejtő kulcsok minőségén múlik. A függvények tervezése és tesztelése évekig is elhúzódó körültekintő munka. Az AES jelölteket például három éven keresztül tesztelték felkért és önkéntes szakértők, mire a NIST[5] 2000 őszén eredményt hirdetett[6].

Egy kriptorendszert ***szimmetrikus***nak nevezünk, ha a titkosító és a megfejtő kulcs azonos, azaz $k_E = k_D$. Ezeket a kriptorendszereket szokás nyílt kulcsúaknak is nevezni. Minden klasszikus titkosító rendszer szimmetrikus. Működésük általában jól átlátható. Egyszerű operációkat használnak, így rendkívül gyorsak.

A titkosító és a megfejtő kulcs azonossága azonban komoly problémát jelent. Hogyan cseréljék ki a kommunikálni szándékozó felek a közös kulcsot? Ameddig a bizalmas üzenetváltás maroknyi katona, forradalmár és kém igénye volt, addig ezt a problémát egy személyes találkozóval meg lehetett oldani. A kulcsot valamilyen félig bizalmas csatornán, például zárt borítékban vagy postagalambbal is el lehet küldeni a partnernek. Utóbbi esetben azonban a kulcs – mint Jules Verne, Sándor Mátyás[7] című könyvében – illetéktelen kezekbe kerülhet.

[5] National Institute of Standard and Technology

[6] Joan Daemen és Vincent Rijmen, The Design of Rijndael: AES - The Advanced Encryption Standard, Springer Verlag, 2002.

[7] Jules Verne, Sándor Mátyás, ford.: Örvös Lajos, Móra Ferenc Könyvkiadó 1969, 1976 és 1980. MEK változat http://mek.oszk.hu/03200/03220/03220.pdf

A kulcscsere (és a digitális aláírás) problémájának megoldására javasolta Diffie és Hellman[8] 1976-ban az ***aszimmetrikus kriptorendszer***ek kidolgozását. A szakirodalomban szokás ezeket nyilvános kulcsú kriptorendszernek is nevezni. Ilyenkor a titkosító és a megfejtő kulcs nemcsak formailag, hanem funkcionálisan is különböznek. A titkosító kulcs nyilvános, bárki megismerheti, vele azonban csak titkosítani lehet. **A titkosított üzenetet kizárólag az tudja dekódolni, aki a megfejtő kulcs birtokában van.** Aszimmetrikus kriptorendszer használatakor tehát $E$, $D$, $E(u, k_E)$ sőt még $k_E$ ismerete sem elegendő $u$ kiszámításához. Paradox módon még az sem tudja dekódolni az üzenetet, aki előtte kódolta.

Az aszimmetrikus kriptorendszer zseniális találmány, nélküle az internet üzleti és igazgatási értéke a töredékére csökkenne. Meg lehet vele oldani a szimmetrikus titkosításhoz szükséges kulcs biztonságos küldését. Ha például Kriszta és Aladár bizalmas üzenetet szeretne váltani, akkor Kriszta ehhez kitalál egy $k$ kulcsot. Aladár nyilvános kulcsával kódolja $k$-t és a kódolt értéket elküldi neki például e-mailben. Az így titkosított értékből csakis Aladár tudja visszaállítani $k$-t. Tehát mindketten, de csak ők, ismerik a közös kulcsot és megkezdhetik a bizalmas üzenetváltást. A kulcscserével részletesebben a 4.3 fejezetben foglalkozunk.

Diffie és Hellman rámutatott arra is, hogy aszimmetrikus kriptorendszerrel dokumentumokat alá is lehet írni, azaz *hitelesíteni* is lehet. A titkos kulcs ugyanis egyértelműen azonosítja a tulajdonosát. Ha tehát az aszimmetrikus titkosítás eljárását megfordítjuk és Aladár a titkos kulcsával kódol egy dokumentumot, majd nyilvánosságra hozza mind a dokumentumot, mind annak aláírt (titkosított) változatát, akkor bárki ellenőrizni tudja, hogy a dokumentum Aladártól származott és hiteles-e. Ehhez a titkosított változatot Aladár nyilvános kulcsával kell dekódolni és összevetni az – ugyancsak Aladártól származó – eredeti változattal. Ha a dekódolt és az eredeti változatok megegyeznek, akkor a dokumentum hiteles, különben a dokumentumot vagy nem Aladár írta alá vagy közben valaki megváltoztatta vagy valami miatt megváltozott. A digitális aláírásról részletesebben a 4.5 fejezetben írunk.

A figyelmes olvasó biztosan kifogásolja, hogy a kriptorendszer definíciójakor fix hosszúságú üzenetblokkok titkosítása szerepelt, ugyanakkor az előző bekezdésekben dokumentumok titkosításáról is írtunk, amelyek hossza tetszőleges lehet. Ezt az ellentmondást könnyen fel lehet oldani. A dokumentumokat manapság számítógépen írjuk. Minden karakternek – kis- és nagybetű, számjegyek, írásjelek, beleértve a szóközt és a sorvéget is – megfelel egy 16 bitből, azaz 2 bájtból, álló bináris jelsorozat. A 'a' betű kódja például 00000000 01100001 = 61, az '1' számjegyé 00000000 00011111 = 31. Ahogy a képernyőn a karaktereket egymás mellé írva szavakat kapunk, amelyekből a mondatok, végül pedig az egész dokumentum kialakul a számítógép a karaktereknek megfelelő kódokat tárolja el egymás után. A dokumentumot tehát egy hosszú bit- vagy bájtsorozat formájában tárolja.

[8]Whitfield Diffie and Martin Hellman, New direction in cryptography, IEEE Trans. on Information Theory, 22 (1976), 644-654.

A titkosító eljárásnak semmi mást sem kell tenni, mint a dokumentumot a számítógépben ábrázoló bitsorozatot megfelelő hosszúságú blokkokra vágni és a blokkokra külön-külön a titkosító függvényt alkalmazni. A megfejtés hasonlóan működik. Először a titkosított blokkokat külön-külön dekódoljuk, az eredményeket pedig sorban egymás mellé írjuk. Ha minden rendben folyik, akkor a folyamat végén, képernyőnkön megjelenik a dekódolt és az ember számára olvasható dokumentum.

Végezetül felhívjuk a figyelmet arra, hogy bizonyos alkalmazások esetén, például jelszavas azonosítás és a – digitális aláírásnál alkalmazott – kivonat készítés során a kriptorendszerben nincs szükség dekódoló függvényre, sőt kulcsra sem. Ilyenkor csak egy olyan titkosító függvény kell, amelyet könnyű kiszámítani, de nagyon nehéz dekódolni. Ezeket *egyirányú* vagy *hash* függvénynek nevezzük. A nyomtatott telefonkönyv jó példa egyirányú függvényre. Ebből nagyon könnyű egy személy telefonszámát kikeresni, de egy telefonszám ismeretében nagyon nehéz a szám tulajdonosát megtalálni. Elektronikus „telefonkönyvnél" persze ilyen probléma nincs, telefonszámhoz ugyanolyan könnyű a tulajdonost megkeresni, mint fordítva.

## *Tesztkérdések:*

T13. Egészítse ki a következő mondatot: Az (U,K,T,E,D) ötöst kriptográfiai rendszernek nevezzük, ha U, K és T _____, az E és a D pedig _____.

T14. Mely állítások igazak a következők közül? Modern, szimmetrikus kriptorendszerekben

a) a titkosítás algoritmusa ismert, de a megfejtésé nem,

b) a titkosító és a megfejtő kulcs azonos és csak a partnerek ismerik,

c) a titkosító kulcs nyilvános,

d) a megfejtés algoritmusa ismert, de a titkosításé nem,

e) mindkét algoritmus szabványos.

T15. Egészítse ki a következő mondatot: Aszimmetrikus kriptorendszerben a titkosító kulcs _____ a megfejtő kulcs pedig _____.

# 2 Szimmetrikus kriptorendszerek

## 2.1 Klasszikus szimmetrikus kriptorendszerek

### 2.1.1 A Caesar kriptorendszer

Bár a kriptográfiát nyugodtan tekinthetjük a modern kor tudományának, ez nem jelenti azt, hogy a titkosítás mint eszköz ne lett volna jelen az emberiség történetének jóval korábbi időszakában. A rómaiak feljegyezték Julius Caesarról, hogy amennyiben bizalmas üzenetet kívánt küldeni, akkor az abécé betűit úgy használta, hogy az első betű helyett a negyediket választotta (azaz az A helyett mindig D betűt írt), a második helyett az ötödiket, és így tovább. Ahhoz, hogy egy ilyen titkosított üzenetet valaki visszafejtsen a negyedik betű helyett az elsőt, az ötödik helyett a másodikat, stb. kellett behelyettesíteni. Így persze az utolsó három betűhöz nem tudnánk mit hozzárendelni, ezért ezekhez rendre az 1., a 2. és a 3. betűket rendeljük hozzá. A titkosítás és visszafejtés megkönnyítése érdekében készíthetünk egy táblázatot, ami a 26 betűs angol abécé használata esetén a következőképpen néz ki:

| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c |

A fenti táblázatot úgy használjuk, hogy titkosításkor megkeressük a titkosítandó betűt a táblázat első sorában, és helyette a táblázat második sorában alatta szereplő betűt írjuk a titkosított szövegbe. Természetesen, ha a második sorban nem 3, hanem $k$ hellyel toljuk el a „betűsort", akkor egy teljesen hasonló titkosító függvényt kapunk, így lényegében $k$ a titkosító kulcsnak tekinthető. A fenti ismertetésből úgy tűnhet, hogy ennek a titkosításnak semmi köze a matematikához. Ez természetesen csak a látszat. Valójában a betűknek számokat feleltethetünk meg az alábbi módon:

| a | b | c | d | e | f | g | h | i | j | k | l | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| n | o | p | q | r | s | t | u | v | w | x | y | z |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

Ha a betűk helyett a számokat használjuk, akkor a nyílt szövegek halmaza és a titkosított szövegek halmaza is a $\{0,1,2,\dots,25\}$ abécé feletti véges szavakból áll.A kulcstér szintén a $K = \{0,1,2,\dots,25\}$ halmaz, és a titkosító függvény (mely az abécé betűin egyenként hat) az

$$E_k(m) := m + k \pmod{26}$$

függvény, míg a visszafejtő függvény a

$$D_k(m) := m - k \pmod{26}.$$

Ezt a kriptorendszert nevezik a ***Caesar-féle kriptorendszer***nek. Nagy hátránya, hogy a kulcstér mérete túl kicsi (26 elemű), így akár próbálgatással is könnyen megfejthető.A kulcstér mérete úgy

növelhető, ha a betűk helyett betűpárokat, vagy betű $n$-eseket tekintünk az abécé elemeinek, és ezeket feleltetjük meg egész számoknak. Megjegyezzük azonban, hogy a Caesar kriptorendszer ezen általánosítása is túl könnyen megfejthető, és a számítógépek korában önmagában egyáltalán nem biztonságos.

### 2.1.2 A helyettesítéses kriptorendszer

A Caesar kriptorendszer minden betű helyett egy másik betűt helyettesít be, mégpedig kölcsönösen egyértelmű módon, így a titkosító függvény lényegében az abécé egy permutációja. Ugyanakkor a kulcstér nagyon kis méretű, azaz az összes lehetséges permutáció közül csak néhány lehet titkosító függvény. A helyettesítéses titkosítás abban az értelemben általánosítása a Caesar-féle kriptorendszernek, hogy ennél minden lehetséges permutáció felléphet titkosító függvényként. Megint az angol abécét alapul véve az alábbi táblázat egy példát mutat a helyettesítéses kriptorendszerre:

| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| y | g | o | f | q | a | b | j | l | m | k | p | x | r | s | w | t | h | n | v | i | e | d | z | c | u |

A helyettesítéses kriptorendszer fő előnye a Caesar kriptorendszerrel szemben a jóval nagyobb kulcstér. Valóban, egy$n$-betűs abécé felett a Caesar kriptorendszer kulcsterének számossága $n$, míg a helyettesítes kriptorendszer kulcsterének számossága $n! = 1 \cdot 2 \cdot \ldots \cdot \mathrm{n}$. Ugyanakkor hátránya a helyettesítéses kriptorendszernek, hogy a használata körülményesebb, és bár a számítógépek korában ez nem jelent igazi problémát, a kulcs bonyolultabb, így azt mindenképpen tárolni kell valahol (a Caesar kriptorendszer esetén fejben megjegyezhető).

A helyettesítéses titkosításnak nagy hiányossága az, hogy a természetes nyelvekben a karakterek (betűk) előfordulásának a gyakorisága igen szigorú szabályoknak felel meg, így a gyakoriságanalízis segítségével, nagy eséllyel megfejthető egy kellően hosszú helyettesítéses kriptorendszerrel titkosított szöveg. A gyakoriságanalízis segítségével egy üzenet megfejtése a következőképpen zajlik:

- A karaktereket három osztályba szokás sorolni: nagy, közepes és kis gyakoriságú.
- A helyettesítéses titkosítás nem módosítja a karakterek előfordulásának a gyakoriságát, hanem csak azt, hogy a betűket milyen jellel jelöltük. Ezek szerint, ha egy kriptoanalitikus kap egy titkosított üzenetet (T), amely elég hosszú és tudja, hogy a természetes nyelvből helyettesítéses titkosítással keletkezett, akkor a következőket teszi:
    - Megpróbálja megállapítani, hogy milyen nyelven készült az eredeti szöveg (Ez lehet, hogy csak az elemzés későbbi lépéseiben alakul ki.)
    - Megállapítja T-ben a karakterek előfordulásának a gyakoriságát. Ez egyszerű leszámlálással és osztással történik. Ebből kiderül, hogy melyek a leggyakoribb karakterek T-ben.

- Kicserélve T-ben a leggyakoribb 2-3 karaktert a természetes nyelvben leggyakoribb karakterekkel megpróbál minél hosszabb összefüggő szövegrészletet azonosítani. Ezzel további karaktereket tud azonosítani a kisebb előfordulású jelek közül.
- Végül megfejti az egész üzenetet.

Manapság hatékony nyelvi elemzők állnak rendelkezésre a dekódolás támogatására.

További lehetőség a helyettesítéses kriptorendszer biztonságosabbá tételére, ha betűk helyett betűpárokat, betűcsoportokat titkosítunk, ám a nyelvi szabályszerűségek a betűpárok, vagy betűcsoportok előfordulási gyakoriságára is kiterjednek, így a gyakoriságanalízis segítségével ekkor is megfejthetők az üzenetek.

Megjegyezzük, hogy bizonyos helyettesítéses kriptorendszerek ma is jól használhatók nem nyelvi jellegű információk, hanem például képek titkosítására.

### 2.1.3 A Vigenère kriptorendszer

A helyettesítéses (és így speciálisan a Caesar) kriptorendszer azért törhető fel könnyen gyakoriságanalízis segítségével, mert a nyílt szöveg egy adott betűjének a titkosító függvény általi képe minden esetben ugyanaz a betű lesz. Ez a tulajdonság akkor sem változik, ha a betűk helyett betűpárokat vagy betűcsoportokat titkosítunk helyettesítéses kriptorendszerrel, így a gyakoriságanalízis ebben az esetben is hatékony eszköz a kriptorendszer feltörésére. A helyettesítéses (és ezen belül az eltolásos) kriptorendszernek ezt a gyenge pontját igyekszik kijavítani a Vigenère kriptorendszer, melyet Blais de Vigenère francia diplomatáról neveztek el.

A Vigenère kriptorendszer bemutatásához is használjuk az angol abécét, és tegyük fel, hogy egy ezen abécé betűiből álló üzenetet szeretnénk titkosítani. Ehhez elkészítjük az alábbi táblázatot:

| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a |
| c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b |
| d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c |
| e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d |
| f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e |
| g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f |
| h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g |
| i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h |
| j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i |
| k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j |
| l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k |
| m | n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l |
| n | o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m |
| o | p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n |
| p | q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o |
| q | r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p |
| r | s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q |
| s | t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r |
| t | u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s |
| u | v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t |
| v | w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u |
| w | x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v |
| x | y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w |
| y | z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x |
| z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y |

A fenti táblázat úgy készült, hogy minden sora tartalmazza az abécé összes betűjét, úgy, hogy az $n$-edik sor az abécé$n$-edik betűjével kezdődik, majd sorra szerepelnek a betűk a z-ig, aztán pedig az a betűtől kezdve sorra jönnek az adott sorban még nem szereplő betűk.

### *A Vigenère kriptorendszer használata:*

- Először elkészítjük a fenti Vigenère-táblázatot.
- Ezután a Vigenère kriptorendszer használatához szükség van egy kulcsra, ami egy az abécé betűiből álló tetszőleges szó.

- Ezt a kulcsszót addig írjuk folytatólagosan újra és újra a titkosítandó szöveg alá, amíg a szöveg minden betűje alá kerül egy-egy betű a kulcsszóból.
- A titkosítandó szöveg minden betűjét a következőképpen helyettesítjük egy másik betűvel:
  - tekintjük a táblázat azon oszlopát, amely a titkosítandó betűvel kezdődik
  - megkeressük azt a sort, amelyik a titkosítandó betű alá írt kulcsszó-betűvel kezdődik
  - a titkosítandó betűnek a titkosító függvény általi képe a fenti oszlop és sor metszetében álló betű lesz.

Ebből a leírásból is jól látszik, hogy a Vigenère kriptorendszer lényegében több különböző kulcsú Caesar típusú kriptorendszer összefésülése, hiszen a nyílt szöveg minden olyan betűjét ami a kulcsszó adott pozícióban lévő betűje fölé esnek, mindig ugyanannyival (mégpedig a kulcsszó adott pozícióban lévő betűjének megfelelő számmal) toljuk el a titkosítás során.

A példa kedvéért titkosítsuk a „Let us meet in the pub at midnight" szöveget, és tegyük fel, hogy a titkosításhoz a „secret" kulcsszót használjuk:

| | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| L | E | T | U | S | M | E | E | T | I | N | T | H | E | P | U | B | A | T | M | I | D | N | I | G | H | T |
| S | E | C | R | E | T | S | E | C | R | E | T | S | E | C | R | E | T | S | E | C | R | E | T | S | E | C |
| D | I | V | L | W | F | W | I | V | Z | R | M | Z | I | R | L | F | T | L | Q | K | U | R | B | Y | L | V |

A táblázat alsó sorában a titkosított szöveg található.

Fontos megjegyezni, hogy a különböző pozícióban lévő azonos betűk titkosító függvény általi képe különbözik. Így az egyszerű gyakoriságanalízis segítségével nem lehet feltörni ezt a rendszert.

### *A Vigenère kriptorendszer visszafejtése:*

- A kulcsszót addig írjuk folytatólagosan újra és újra a titkosított szöveg fölé, amíg a titkosított szöveg minden betűje alá kerül egy-egy betű a kulcsszóból
- A titkosított szöveg soron következő betűjét megkeressük abban a sorban, amelyik az ezen betű feletti kulcsszó-betűvel kezdődik, és megnézzük melyik oszlopban található
- Az eredeti nyílt szöveg megfelelő betűje ennek az oszlopnak az első betűje volt.

### *A Vigenère kriptorendszer visszafejtése, kriptoanalízise:*

A Vigenère kriptorendszert sokáig feltörhetetlennek tartották, mígnem 1863-ban F.W. Kasiskinek sikerült olyan módszert találnia, amivel a Vigenère kriptorendszer feltörhető.

Kasiski módszerének alapötlete a következő: a titkosított szövegben található azonos blokkok nagy valószínűséggel a kulcsszó azonos szakaszához tartoznak. Tehát, ha vesszük a titkosított szövegben található azonos blokkok távolságainak legnagyobb közös osztóját, akkor a különböző ismétlődő blokkokhoz tartozó lnko-k leggyakrabban előforduló legnagyobb faktora valószínűleg a kulcsszó

hosszát adja. Ezután egy erre az információra alapozott finomított gyakoriságanalízis segítségével a Vigenère kriptorendszer könnyen feltörhető.

*Tesztkérdések:*

T16. Egy Caesar-típusú titkosító függvény (mely a 26 betűs angol abécé betűin van értelmezve) az A betűnek az F betűt felelteti meg. Mit feleltet meg a C betűnek?

T17. Egy Caesar-típusú titkosító függvény (mely a 26 betűs angol abécé betűin van értelmezve) az A betűnek az L betűt felelteti meg. Melyik betűnek a titkosító függvény általi képe az R betű?

T18. Az alábbiak közül melyek azok a gyenge pontok, amelyek miatt egy Caesar-kriptorendszer nem tekinthető biztonságosnak?

a) Kicsi a kulcstér.
b) Védtelen a „man in the middle" típusú támadással szemben.
c) A titkosított szöveg hossza duplája a nyílt szöveg hosszának.
d) Gyakoriságanalízis segítségével könnyen megfejthető.

T19. Hány különböző kulcsa lehet egy 26 betűs abécén definiált Caesar-típusú kriptorendszernek?

a) $26 \cdot 25$
b) $26$
c) $26! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot 26$
d) $\binom{26}{2}$

T20. Hány különböző kulcsa lehet egy 26 betűs abécén definiált helyettesítéses kriptorendszernek?

a) $26 \cdot 25$
b) $26$
c) $26! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot 26$
d) $\binom{26}{2}$

## 2.2 Modern szimmetrikus kriptorendszerek

### 2.2.1 One time pad

A one time pad vagy Vernam titkosítót magyarul egyszer használatos bitmintának is nevezik. Mi azonban maradunk az általánosan elfogadott angol terminológiánál. Digitális üzenetek titkosítására lehet használni, ami nem jelent lényeges megszorítást, hiszen bármilyen üzenetet át lehet kódolni digitális formába. A one time pad lényege az, hogy adott üzenethez elkészítünk egy ugyanolyan hosszúságú véletlen bitsorozatot, majd ezt bitenként xorozzuk az üzenettel. A titkosító kulcs szerepét ebben az esetben a véletlen bitsorozat játsza.

Az xor művelet, amelyet⊕-vel jelölünk egyforma bitekhez 1-et, különbözőekhez pedig 0-t rendel. A kriptográfiában gyakran használják, mert rendelkezik azzal a tulajdonsággal, hogy ha az*a* bithez xorozzuk a *b* bitet, majd az eredményhez ismét xorozzuk a *b* bitet, akkor visszakapjuk az *a* bitet. Mindezt képletben így fejezhetjük ki $(a \oplus b) \oplus b = a$. A tulajdonság könnyen bizonyítható. Ha például *b = 1*, akkor $a \oplus b = a$ és így $(a \oplus b) \oplus b = a \oplus b = a$.

A one time pad-el kódolt üzenetből úgy kapjuk tehát vissza az eredetit, hogy a tikosító kulcsot bitenként xorozzuk a titkosított üzenettel. A megfejtő kulcs tehát megegyezik a titkosító kulccsal, ezért a továbbiakban csak kulcsot írunk.

A one time pad nagyon biztonságos titkosító eljárás, mert a kódolt szöveget a kulcs ismerete nélkül csak az összes lehetséges adott hosszúságú bitsorozat kipróbálásával lehet megfejteni. Ehhez természetesen a kulcsnak valóban véletlen és csak egyszer használható bitsorozatnak kell lenni. Mindkét feltétel nehezen teljesíthető egyszerre, mert megfelelő mennyiségben csak álvéletlen sorozatokat tudunk generálni.

A one time pad igazi gyenge pontját az egyszer használatos kulcs megosztása jelenti a küldő és fogadó felek között. Az üzenettel megegyező hosszúságú bitsorozatot kell biztonságos úton eljuttatni a fogadó félnek. Ha ezt meg tudjuk oldani, akkor persze ugyanezen az úton az eredeti üzenetet is el tudjuk juttatni neki biztonságosan. Ezért a one time pad-et a gyakorlatban nem használják, elméleti szempontból azonban fontos.

### 2.2.2 DES

Évszázadokon keresztül a jelentősebb méretű adatok továbbítását is lehetővé tevő titkos, biztonságos kommunikáció igénye lényegében a katonai és diplomáciai körök sajátja volt, a mindennapi életben ilyen jelentős igény nem jelentkezett. Ugyanakkor a második világháború után olyan társadalmi folyamatok zajlottak le, és (a számítógépek megjelenésével a hétköznapokban) olyan technológiai fejlődésnek lehettünk tanúi, aminek eredményeképpen a biztonságos, titkos kommunikáció igénye a civil szférában, elsősorban az üzleti életben is megjelent. Így szükségessé vált egy olyan kriptorendszer kidolgozása, melyet a civil szféra szereplői korlátozás nélkül használhatnak, éspedig úgy, hogy a kommunikáció során ne legyen szükség méretes, bonyolult titkosító-berendezések használatára, hanem a titkosítás számítógépes program segítségével történhessen. Ezt a szükségletet kielégítendő az USA Szabványügyi Hivatala (National Buro of Standards, ma National Institute of Standards and Technology) 1973-ban javaslatot kért egy kriptográfiai algoritmus szabványára, az alábbi alapvető elvárásokkal:

- Magas biztonsági szint.
- Teljes, könnyen érthető leírás.
- Az algoritmus biztonsága csak a kulcstól függhet, nem az algoritmus titkosságától.
- Minden felhasználó számára elérhető.
- Különféle felhasználási formákhoz jól igazítható.

- Gazdaságosan implementálható elektronikai eszközökön.
- Hatékony.
- Ellenőrizhető.

Az elfogadott szabvány a DES (Data Encryption Standard) néven lett ismert. A következőekben röviden ismertetjük a DES algoritmust. Ehhez először a Feistel-titkosítókat kell megismernünk, mert a DES is egy ilyen típusú titkosító függvény.

### Feistel-titkosítók

Egy Feistel-titkosító egy olyan titkosító függvény, amely az alábbi módon épül fel:

- Adott egy $\{0,1\}$ abécé feletti $t$ blokkhosszúságú blokktitkosító, melynek $K$ kulcshoz tartozó titkosítófüggvénye $f_K$.
- Rögzítünk egy *r* menetszámot.
- Megadunk egy módszert, amellyel egy $K$ kulcshoz hozzárendelünk $r$ darab menetkulcsot ($K_i$), melyek a belső blokktitkosító kulcsteréhez tartoznak.
- A Feistel-titkosító $k$ kulcshoz tartozó $E_k$ titkosító függvénye az alábbi módon működik:
  - Legyen adott egy $2t$ hosszúságú nyílt szöveg.
  - $P = (L_0, R_0)$, ahol $L_0$ a nyílt szöveg bal fele, $R_0$ pedig a jobb fele.
  - Ezután az alábbi rekurzió szerint készítünk egy sorozatot:

$$(L_i, R_i) = \left(R_{i-1}, L_{i-1} \oplus f_{K_i}(R_{i-1})\right), \quad i = 1, \dots, r.$$

  - Végül legyen

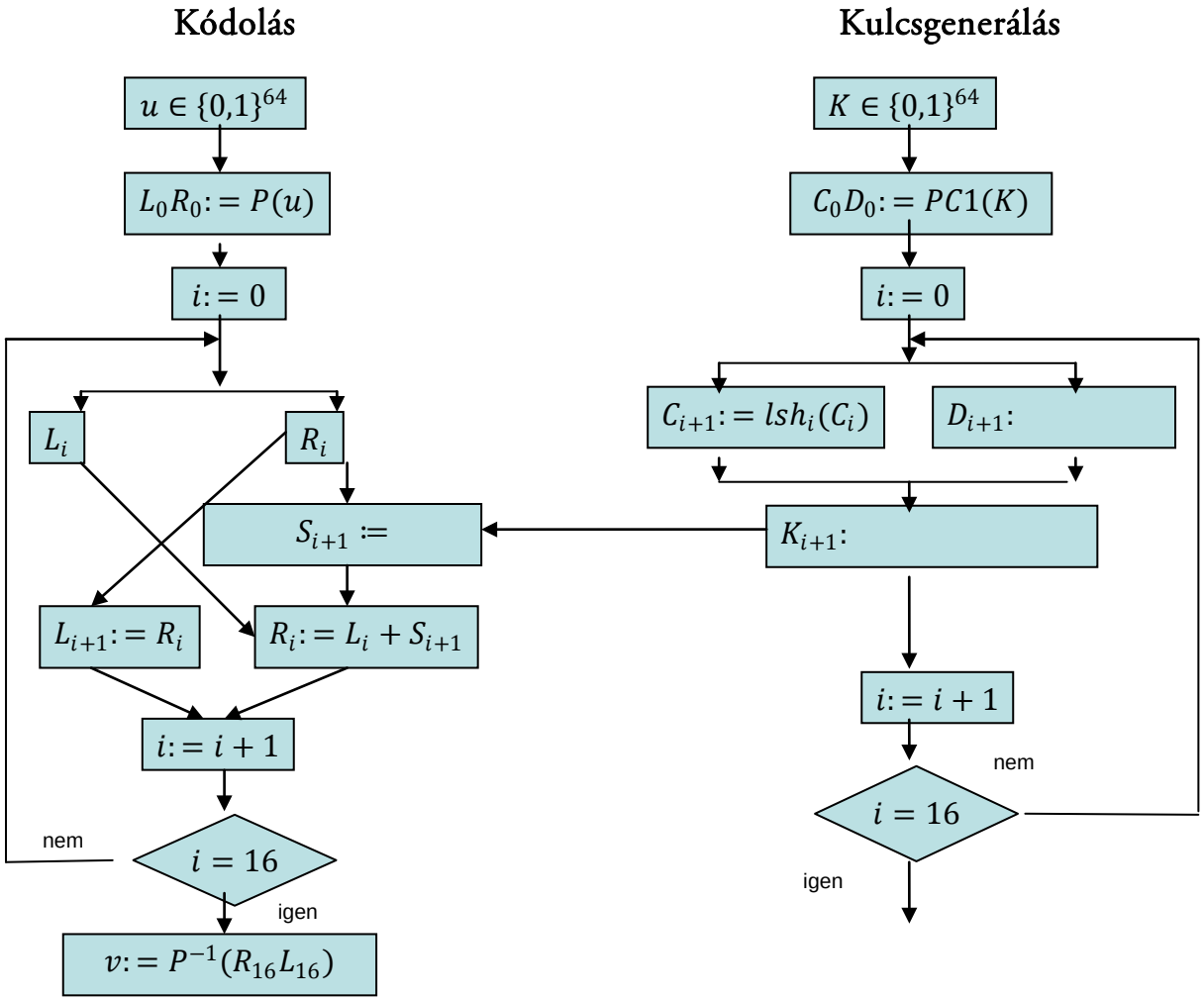
$$E_K(L_0, R_0) = (L_r, R_r).$$

Az algoritmusban $\oplus$ a bitenkénti „kizáró vagy” műveletet jelenti, amelynek értéke $0$, ha a két bit különbözik, és $1$, ha megegyezik.

Megjegyezzük, hogy a Feistel-típusú titkosítókat elsősorban az $f_K$ belső titkosító függvény különbözteti meg egymástól, és a titkosítás biztonsága is elsősorban ezen múlik.

## A DES folyamatábrája, és a DES leírása

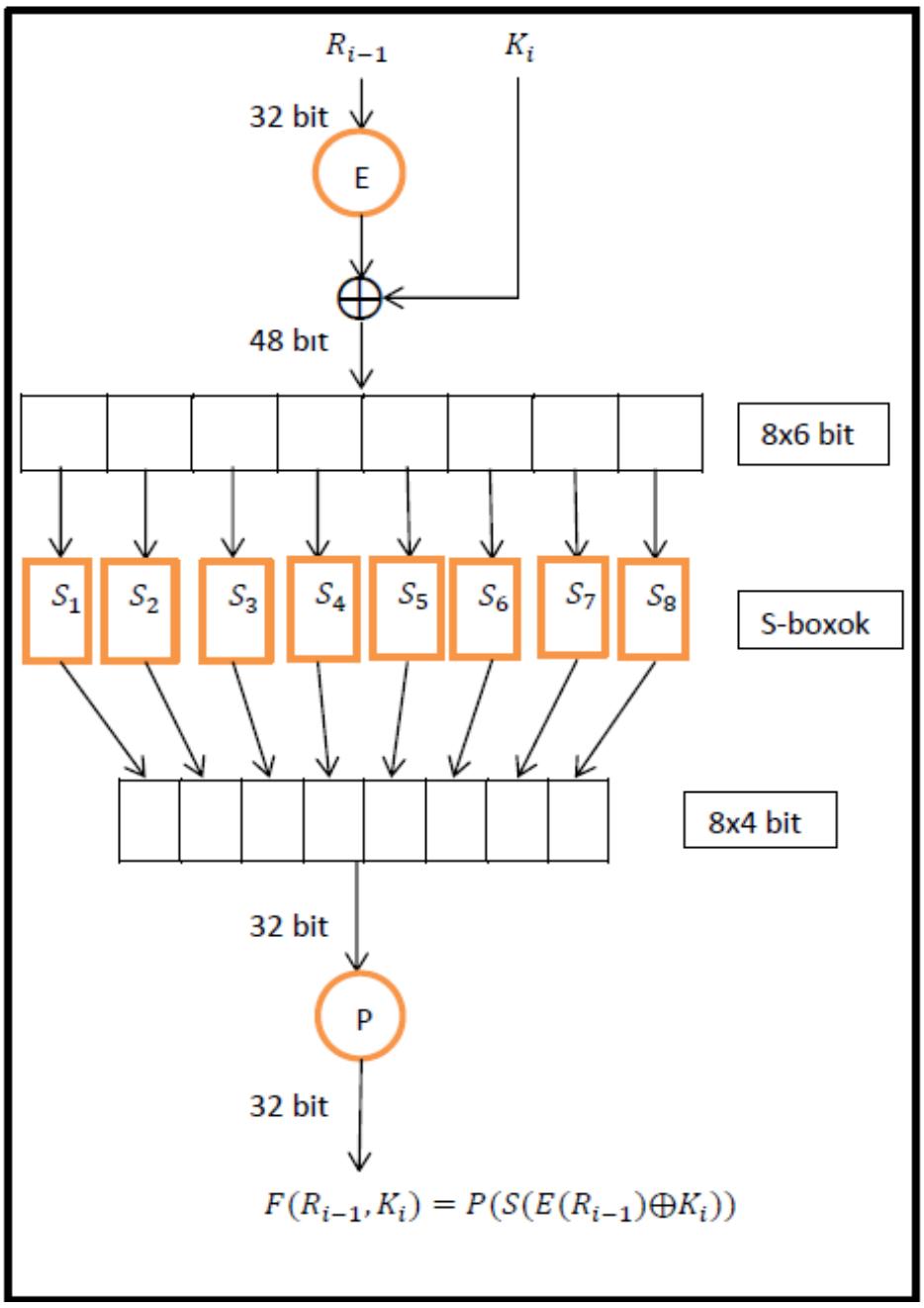
A DES röviden az alábbi folyamatábra segítségével ismertethető:




A fenti folyamatábrából jól látszik, hogy az első lépés egy kezdeti permutáció alkalmazása, majd az így kapott 64-bites blokkot felosztja a bal ($L_0$) és jobb ($R_0$) felére. Ezután 16 meneten keresztül alkalmazni kell a DES menet-transzformációját, végül a kezdeti permutáció inverzének alkalmazása következik. A Feistel-típusú részen belül, a belső titkosító függvény az $F(R_i, K_{i+1})$, mely függ a $K_{i+1}$ menetkulcstól. Most ennek a belső titkosító függvénynek az ismertetése következik.

### *A DES belső titkosító függvénye:*

Az $F(R_i, K_{i+1})$ függvény működését az alábbi folyamatábra összegzi:




Ebben az $E$ egy expanziós függvény, mely a 32 bites bemeneti blokkból egy 48 bites kimeneti blokkot generál. Ehhez a kimenethez bitenként hozzá-XOR-oljuk (kizáró vagy műveletet végzünk bitenként) a szintén 48 bites menetkulcsot, majd az ennek eredményeként adódó 48 bitet 8 darab 6 bites blokkra vágva kapjuk a nyolc S-box bemenetét. Ezek mindegyike 4 bites kimenettel rendelkezik, így újra 32 bites blokkot kapunk, amire még alkalmazzuk a $P$ permutációt, így megkapva az $F(R_i, K_{i+1})$ függvény értékét.

Az ***S-boxok szerepe*** különösen fontos minden modern kriptorendszerben, ugyanis ezek feladata azt garantálni, hogy a titkosító függvény minél távolabb álljon a lineáris függvénytől, ugyanis a lineáris függvénnyel való titkosítás nem tekinthető biztonságosnak.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |

***A DES mesterkulcs*** egy olyan véletlen 64 bites blokk kell legyen, ahol minden nyolcadik bit úgynevezett paritás bit, azaz ha hozzáadjuk az előtte álló 7 bit összegéhez, mindig páratlan szám kell kijöjjön. Így például egy érvényes DES-kulcs lehet a mellékelt táblázatban szereplő 64 bit. Ezt hexadecimálisan felírva az alábbi kulcsot kapjuk:

B3913C6832AE9276

A DES mesterkulcsból az algoritmus futása során 16 darab 48 bites menetkulcsot generálunk, és ezeket használjuk a DES 16 menete során a belső titkosító függvényhez kulcsként.

### *A DES visszafejtése*

Ha a titkosított szövegre a DES algoritmust alkalmazzuk, de a menetkulcsokat fordított sorrendben használjuk, akkor az eredeti üzenetet kapjuk vissza.

### *A TDES*

Az évtizedek során a számítástudomány és a számítógépek fejlődése olyan iramban történt, hogy az 1990-es évekre a DES kezdett kevéssé biztonságossá válni, ugyanakkor a DES kulcshosszának bővítése a szabvány jelentős megváltoztatása nélkül nem volt lehetséges. Így a biztonság növelése érdekében a DES azon tulajdonságát használták ki, hogy a különböző DES kulcsokkal való titkosítás nem alkot félcsoportot a kompozícióra nézve. Ez azt jelenti, hogy, ha a nyílt szöveget egy $K_1$ kulcsot használva a DES segítségével titkosítunk, majd a titkosított szöveget egy $K_2$ kulcsot használva a DES segítségével újra titkosítjuk, akkor nem feltétlenül létezik (sőt, a legtöbb esetben nem létezik) olyan $K$ DES kulcs, mely segítségével a nyílt szövegből a második titkosítás utáni titkosított szöveget egyetlen DES titkosítással megkapjuk. Ezt az algoritmust nevezték double-DES-nek, és amikor már ennek a biztonságossága is megkérdőjeleződött, akkor áttértek a triple-DES használatára, ami három egymás után elvégzett DES-szel való titkosítást jelent. Ennek azonban komoly hátránya, hogy immár háromszor akkora időt igényel a használata, mint a DES használata.

### 2.2.3 AES

Mivel már látható volt, hogy a DES hosszú távon nem felel meg a modern társadalom által támasztott biztonsági elvárásoknak, 1997-ben az NIST (National Institute of Standards) pályázatot írt ki az AES (Advanced Encryption Standard) megalkotására. A kiírás szerint egy

- nyilvánosan közzétett,
- díjmentes,
- világszerte terjesztett,
- a következő évszázadban a kormányzati dokumentumok védelmére alkalmastitkosító algoritmus megalkotását várták el, mely feleljen meg az alábbi kritériumoknak:
- szimmetrikus kriptorendszert valósítson meg,
- blokkódoló legyen,
- kezelje a 128-128, 128-192 és a 128-256 bites blokkhossz-kulcshossz kombinációt,
- legyen legalább olyan biztonságos, mint a TDES,
- legyen sokkal gyorsabb, mint a TDES,
- a tervezőnek le kell mondania a szerzői jogokról.

A pályázatok beadási határideje 1998. június 15. volt. Határidőig 15 pályázat érkezett, melyek közül az első bírálati kört 5 javaslat élte túl: MARS, RC6, Rijndael, Serpent, Twofish. A döntés végül 2000. október 2.-án született meg: a 128 bites Rijndael lett az AES. Mivel a Rijndael a 192 bites és 256 bites blokkhossz és kulcshossz kezelésére is alkalmas, és mivel várhatóan később a nagyobb blokkhosszal és kulcshosszal működő variáns is felhasználásra kerülhet, így érdemes általánosabban, a Rijndael algoritmust bemutatnunk.

#### A Rijndael leírása:

A Rijndael név a tervezői, Joan Daemen és Vincent Rijmen nevéből lett összerakva.
A tervezés során figyelembe vett fő szempontok:

- ellenállás az ismert támadásokkal szemben
- gyorsaság és tömör kódolhatóság a platformok minél szélesebb körén
- egyszerű kivitel
- a Rijndael blokkhossza: 128, 192, 256 bit
- a Rijndael kulcshossza: 128, 192, 256 bit
- mindegyik blokkhossz mindegyik kulcshosszal használható

A Rijdael nem Feistel struktúrájú blokktitkosító, mivel azokban az egyes menetek során a bitek legnagyobbrészt csak összekeverednek, de nem változnak meg a bitek. A Rijndael menet-transzformációja ezzel szemben három invertálható rétegből áll

- lineáris keverő réteg (ez teríti szét az információt a blokk teljes hosszában)

- nem-lineáris réteg (egy S-boksz, azaz a lineáristól minél távolabb álló transzformáció)
- kulcshozzáadó réteg (a menetkulcs egyszerű XOR-ozása az aktuális állapothoz)

### *A Rijndael működésének részletes ismertetése:*

A Rijndael 128, 192, illetve 256 bites blokkok kódolására képes 128, 192, illetve 256 bites kulcsok felhasználásával. Legyen $N_b$ a blokkhossz 32-ed része, $N_k$ pedig a kulcshossz 32-ed része Az ***állapot (state)*** a titkosítás köztes eredménye, ami mindig egy a blokkhosszal megegyező méretű bitsorozat. Ekkor az állapotot ábrázolhatjuk egy 4 sorból és Nb oszlopból álló táblázatként, ahol a táblázat minden eleme egy byte. Hasonló ábrázolás használható a kulcs esetén. Az alábbi táblázatok a 128 bites blokkhossz és 128 bites kulcshossz (azaz az AES) esetét mutatják be:

| | | | |
|---|---|---|---|
| $a_{0,0}$ | $a_{0,1}$ | $a_{0,2}$ | $a_{0,3}$ |
| $a_{1,0}$ | $a_{1,1}$ | $a_{1,2}$ | $a_{1,3}$ |
| $a_{2,0}$ | $a_{2,1}$ | $a_{2,2}$ | $a_{2,3}$ |
| $a_{3,0}$ | $a_{3,1}$ | $a_{3,2}$ | $a_{3,3}$ |

| | | | |
|---|---|---|---|
| $k_{0,0}$ | $k_{0,1}$ | $k_{0,2}$ | $k_{0,3}$ |
| $k_{1,0}$ | $k_{1,1}$ | $k_{1,2}$ | $k_{1,3}$ |
| $k_{2,0}$ | $k_{2,1}$ | $k_{2,2}$ | $k_{2,3}$ |
| $k_{3,0}$ | $k_{3,1}$ | $k_{3,2}$ | $k_{3,3}$ |

A kezdeti állapot feltöltése a nyílt szöveggel az alábbiak szerint történik:

- Az inputot 8 bites byte-ok alkotják, melyek 0-tól $(4 \cdot N_b - 1)$-ig vannak sorszámozva.
- Ha ebben a számozásban a byte sorszáma $n$, akkor a táblázatbeli kétdimenziós $(i, j)$ indexe az alábbi összefüggésekből számítható ki:

$$i = n \pmod 4 , \quad j = \left\lfloor \frac{n}{4} \right\rfloor , \quad n = 4 \cdot j + i.$$

A menetek $N_r$ száma a blokkhossz és a kulcshossz függvényében van megadva:

| $N_r$ | $N_b = 4$ | $N_b = 6$ | $N_b = 8$ |
|---|---|---|---|
| $N_k = 4$ | 10 | 12 | 14 |
| $N_k = 6$ | 12 | 12 | 14 |
| $N_k = 8$ | 14 | 14 | 14 |

A menet-transzformáció az alábbi módon néz ki:

```
Round(State, RoundKey)
        {
        ByteSub(State);
        ShiftRow(State);
        MixColumn(State);
        AddRoundKey(State,
        RoundKey);
        }
```

Az utolsó menet kicsit különbözik a többitől

```
        Round(State, RoundKey)
        {
        ByteSub(State);
        ShiftRow(State);
        AddRoundKey(State,
        RoundKey);
        }
```

### *A Rijndael menet-transzformációjának alkotórészei:*

A Rijndaelt alkotó különböző függvények matematikai hátterűek; nagymértékben támaszkodnak a $2^8$ elemű véges test tulajdonságaira, illetve az e feletti polinomgyűrű tulajdonságaira. Ennek a matematikai háttérnek nagy előnye az, hogy ezek a transzformációk matematikai módszerekkel jól elemezhetők, vizsgálhatók. Ugyanakkor a transzformációk nagyrészt leírhatók egyszerű bitműveletek formájában is, ami nagyban segíti a hatékony programozhatóságot:

- A ***ByteSub*** transzformáció egy S-box, ami
  - az állapot egy-egy byte-ján külön-külön hat
  - egy invertálható transzformáció, mely két részből áll:
    - először vesszük az inverzet a $2^8$ elemű testben,
    - majd alkalmazunk egy affin transzformációt
  - - bár ezen S-box mögött komoly matematikai háttér van -, a kimenete minden byte esetén előre kiszámítható, és táblázatba foglalható
  - A ByteSub inverze is egy ByteSub és ugyanannyi időbe telik elvégezni
  - A ByteSub és a ShiftRow sorrendje felcserélhető
- A ***ShiftRow*** transzformáció az állapot sorait shifteli balra ciklikusan. Az $i$-edik sort $C_i$-vel shifteljük, ahol $C_i$ függ a blokkhossztól:

| $N_b$ | $C_0$ | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|---|
| 4 | 0 | 1 | 2 | 3 |
| 6 | 0 | 1 | 2 | 3 |
| 8 | 0 | 1 | 3 | 4 |

- A ***MixColumn*** transzformáció során az állapot egy-egy oszlopát (azaz 4 bájtot) alakítjuk át:
  - a háttérben a $2^8$ elemű véges test feletti polinomokkal végzett műveletek állnak, de a MixColumn bármely 4 bájthoz tartozó kimeneti értéke előre kiszámítható és táblázatba foglalható
  - a MixColumn inverze egy vele azonos struktúrájú transzformáció
- Az ***AddRoundKey*** transzformáció az állapot és a menetkulcs bitenkénti összeadása, azaz az állapot és a menetkulcs megfelelő bitjeire a kizáró vagy logikai műveletet alkalmazzuk

### *Menetkulcs-generálás:*

- A szükséges menetkulcsok bitjeinek száma $32 \cdot N_b \cdot (N_r + 1)$ .
- Először a titkosító kulcsból egy kiterjesztett kulcsot generálunk, amely $N_b \cdot (N_r + 1)$ darab 4 byte-os szóból áll.
- Ebből választjuk ki a menetkulcsokat úgy, hogy az első menetkulcs a kiterjesztett kulcs első $N_b$ darab 4 byte-os szava, a második menetkulcs a második $N_b$ darab, stb.

*A Rijndael titkosító algoritmus:*

- Először alkalmazunk egy kulcshozzáadást,
- Majd jön $N_r - 1$ menet,
- Végül egy utolsó menet (amelyik kicsit különbözik a többitől).

*Tesztkérdések:*

T21. Hány bites blokkokat titkosít a DES?
   a) 128
   b) 56
   c) 64
   d) 256

T22. Hány hasznos bit van egy DES mesterkulcsban?
   a) 56
   b) 128
   c) 192
   d) 256

T23. Hány menetből áll a DES?
   a) 10
   b) 12
   c) 16
   d) 32

T24. A DES alább felsorolt transzformációi közül melyiknek a feladata garantálni, hogy a DES transzformáció minél jobban különbözzön a lineáris transzformációktól?
   a) A kezdeti permutáció.
   b) Az S-boxok.
   c) Az E expanziós függvény.
   d) A P permutáció

T25. Az alábbi titkosító függvények közül melyik Feistel-típusú titkosító?
   a) AES
   b) RSA
   c) DES
   d) Massey-Omura
   e) ElGamal

T26. Miért vált szükségessé a TDES használata?
   a) Mert a DES lassú.
   b) Mert a DES önnmagában már nem volt elég biztonságos.
   c) Mert szerzői jogi problémák adódtak a DES használatával.

T27. Miért biztonságosabb a TDES a DES-nél?

a) Mert a különböző DES kulcsokkal való titkosítás nem alkot félcsoportot a kompozícióra nézve.
b) Mert a különböző DES kulcsokkal való titkosítás egymásután elvégezve kiváltható egyetlen DES kulccsal való titkosítással.
c) Mert a TDES-szel való titkosítás gyorsabb, mint a DES-szel való titkosítás.
d) Mert a TDES-szel való titkosítás lassúbb, mint a DES-szel való titkosítás.

T28. Melyik kriptorendszer volt az AES előtt az USA-beli titkosítási szabvány?
a) DES
b) RSA
c) ElGamal
d) Rijndael
e) Massey-Omura

T29. Melyik kriptorendszer egy variánsát választotta az NIST szabványnak 2000-ben?
a) MARS
b) RC6
c) Serpent
d) Twofish
e) Rijndael

T30. Hány bites blokkhossz-kulcshossz kombináció választásával használható az AES?
a) Kizárólag 128 bites blokkhossz és 128 bites kulcshossz választható.
b) 128 bites blokkhosszhoz a 128, 192 és 256 bites kulcshossz bármelyike választható.
c) 128, 192 és 256 bites blokkhossz és 128, 192 és 256 bites kulcshossz tetszőleges kombinációja használható.

T31. Az AES alább felsorolt transzformációi közül melyik az S-box?
a) ByteSub
b) ShiftRow
c) MixColumn
d) AddRoundKey

T32. Milyen típusú transzformáció a ByteSub inverze?
a) ShiftRow
b) MixColumn
c) AddRoundKey
d) ByteSub

T33. Mennyi az AES menetszáma a 128 bites blokkhossz és 128 bites kulcshossz esetén
a) 8
b) 10
c) 12
d) 14

# 3 Azonosítás; példa egyirányú függvény alkalmazására

## 3.1 Az azonosításról általában

Személyek azonosítása, megnevezése csak néhány száz éve vált általánossá, de a viszonylag rövid idő ellenére az azonosíthatóság igénye mélyen beívódott az emberekbe. Robert Merle Madrapur című könyvének főhőse Mr. Vladimir Sergius a következőket gondolja, amikor elveszik az útlevelét: „És főképp mivel magyarázom azt a kínzó és a lelkem mélyén erős nyomot hagyó érzést, hogy az útlevelemmel együtt a személyazonosságomat is elvesztettem? Nem tudom megfejteni ezt a lelkiállapotot. Csak körülírni. És ha jól meggondolom, nem is olyan képtelenség, mert *aki nem tudja igazolni a többi ember előtt, hogy ki, rögtön semmivé válik, elmerül a sokmilliós egyforma tömegben*." ***Az azonosítás annyit jelent, hogy valamiféle bizonyítékát adjuk annak, hogy azonosak vagyunk önmagunkkal, pontosabban azzal, akinek állítjuk magunkat***. Az azonosítás elengedhetetlen kelléke a bizonyíték, amelyet azonosítónak is szokás nevezni.

A társadalmi struktúrák bonyolultságának növekedésével egyenes arányban nőtt azoknak az élethelyzeteknek a száma is, amelyekben igazolni kell magunkat. Íme egy csokorra való a fontosabbak közül:

- ***Közösséghez való tartozás bizonyítása***. A kapcsolat lehet tőlünk független, tartós, akár életünk végéig tartó, mint az állampolgárság, amelyet a személyi igazolvány és az útlevél igazol. Lehet azonban tőlünk függő is, mint a pártok, szakmai társaságok és érdekérvényesítő szervezetek vagy klubok, stb. által kiadott azonosítók.
- ***Képesség bizonyítása***. Például gépjárművezetői engedély, érettségi bizonyítvány, diploma, nyelvvizsga bizonyítvány, stb. Bizonyítványokat rendszerint rövidebb-hosszabb tanulási folyamatot lezáró képességfelmérő vizsga után adnak ki.
- ***Szolgáltatás igénybevétele***: bank-, hitel- és városkártyák, bérletek, stb.
- ***Védett térbe való belépés***. Sok alkalmazottat foglalkoztató szervezeteknek biztosítani kell azt, hogy csak azok léphessenek be az épületbe vagy munkahelyre, akik ott dolgoznak vagy engedélyt kaptak erre. Az ilyen típusú igazolványhoz már jogosultságok is tartoznak, így átmenetet képeznek a hagyományos igazolványok és az informatikai rendszerek azonosítói között.
- ***Informatikai rendszerekhez való hozzáférés***. Tulajdonképpen ez is egy védett tér, azonban nem valós, hanem virtuális tér. A virtuális tér egyre bővül és struktúrája is egyre bonyolultabbá válik. Ma már nemcsak az intézmények és vállalkozások védik a virtuális terüket, hanem az internetes játékok, a chatszobák és közösségi portálok – például facebook, iwiw, myspace – is.

Az azonosítást igénylő élethelyzetek elszaporodásával az azonosítási technikák száma megnőtt és az azonosítók formája is jelentősen megváltozott. Amíg 50 évvel ezelőtt szinte minden bizonyítvány kis, keményfedelű könyvecske volt, ma már a plasztikkártyák uralkodnak. Az azonosítók fajtájuk szerint az alábbi csoportokba sorolhatóak:

- *Birtoklás alapúak* a különböző igazolványok, pecsétek, jelvények, USB eszköz, token, stb.
- *Viselkedés alapú* az aláírás, kézírás, beszédhang, stb.
- *Biometrikus* a fénykép, ujjlenyomat, szem (írisz, retina) geometriája, illat, DNS, stb.
- *Tudás alapú* az 'anyja neve', 'születési idő', jelszó, PIN kód, aszimmetrikus kriptorendszerek titkos kulcsa.

A csoportosítás kicsit önkényes, mert az egyes osztályok között folytonos az átmenet. Például egy olyan tollal készített aláírás, amelyik mikroérzékelőkkel figyeli az írás dinamikáját és geometriáját besorolható a viselkedés alapú és a biometrikus azonosítók közé is. Az aszimmetrikus kriptorendszerek titkos kulcsát lehetetlen megjegyezni, az csak valamilyen számítógéppel olvasható eszközön tárolható.

Régóta tudjuk, hogy az azonosítás hatékonysága és biztonsága lényegesen növelhető több, független azonosító alkalmazásával. Az igazolványokba például a fénykép mellé rendszerint a tulajdonos aláírása és más azonosításra alkalmas adat is bekerül. Utóbbira jó példa a „hungaricumnak" számító 'anyja neve'. A modern technológia lehetőségeit kihasználó több csatornás azonosításra nagyon jó példa, hogy a banki rendszerek internetes bankolás közben a tranzakció végrehajtása előtt egy egyedi kódot küldenek szöveges üzenetben és a tranzakciót csak a kód visszaküldése után fejezik be. A két független csatorna: internet és mobiltelefon használata lényegesen növeli a bankolás biztonságát. Persze, ha valakitől ellopják a netbankolásra alkalmas mobiltelefonját, akkor mindkét csatorna a rabló ellenőrzése alá kerül.

A számítástechnikában az azonosítás azért fontos, mert ezen keresztül jut be a felhasználó a virtuális térbe és fér hozzá a számítógép erőforrásaihoz. Azonosítás után kezdhetjük meg munkánkat, intézhetjük ügyeinket, vagy csak bámészkodhatunk jogosultságainknak megfelelően. Az identitás- és jogosultságkezelés rokon tevékenység, néha össze is mossák őket. Pedig mind funkciójukban, mind technológiájukban teljesen különbözőek.

Az *identitáskezelés* azonosítást jelent és leggyakrabban kriptográfiai vagy biometriai algoritmusokkal operál. A *jogosultságkezelés* ezzel szemben az ügyviteli folyamat része, azt határozza meg, hogy a felhasználók - szerepüktől függően – mennyi erőforráshoz juthatnak hozzá. Jól megtervezett informatikai rendszerekben az identitás- és jogosultságkezelés hézagmentesen együttműködik.

### *Tesztkérdések:*

T34. Milyen élethelyzetekben szükséges az azonosítás?

a) Számítógép bekapcsolásakor.
b) A facebook-ba való belépéskor.
c) Ha készpénzzel fizetek egy élelmiszerboltban.
d) Ha élni akarok a szavazati jogommal.
e) Ha bemegyek az okmányirodába.

T35. Húzza alá azokat a szavakat, amelyek azonosító fajtákat jelentenek: pettyes, viselkedés alapú, tudás alapú, téglalap alakú, biometrikus, fényképes.

T36. Az identitáskezelés és a jogosultságkezelés
   a) szorosan együttműködik,
   b) semmi közük egymáshoz,
   c) szinonim fogalmak.

T37. A _____ azonosítás azt jelenti, hogy az _____ több, _____ azonosítót kér. Egészítse ki az előbbi hiányos mondatot az alábbi szavakkal: független, többcsatornás, ellenőr.

## 3.2 Tudás alapú azonosítás

Az ember-gép interakcióban ez a legelterjedtebb azonosítási fajta. Lényegében a katonaságnál ősidők óta használt jelszavas azonosítás számítógépekre adaptált változata. A katonai verzióban az őrszolgálat éjszakánként egy jelszót kap, amelyet **minden** tag ismer. Az őr csak olyan személyeket enged a tábor közelébe, akik tudják az aktuális jelszót. A számítógépre adaptált változatban a jelszót csak a tulajdonosa ismerheti.

A tudás alapú azonosításnak többféle változata van, terjedelmi okok miatt azonban itt csak a jelszavas azonosítással foglalkozunk. Amíg a felhasználónak egy alkalmazáshoz csak egy jelszót kell ismernie, addig a számítógépnek sok – esetenként százezres vagy milliós nagyságrendű – felhasználó jelszavát kell megjegyeznie és karbantartania. A jelszavak tárolása egy adatbázis állományban történik. Az állomány bejegyzései tartalmazzák a felhasználói nevet, a jelszót, esetleg más azonosítókat is. A jogosultságokat vagy közvetlenül a jelszóadatbázisban vagy egy ahhoz szorosan kapcsolódó adatbázisban tárolják.

Felhívjuk a figyelmet arra, hogy megfelelő jogosultsággal a számítógépen tárolt bármilyen állomány olvasható, sőt módosítható is. A jelszóadatbázis birtoklása széles körű hozzáférést biztosít a számítógép(ek) erőforrásaihoz, ezért a hackerek kedvelt célpontja. Úgy kell tehát abban az adatokat tárolni, hogy még ha valaki illetéktelenül olvassa is az adatbázist ne tudjon más felhasználót megszemélyesíteni. Más szavakkal ez annyit jelent, hogy ***egy hacker elolvasva a jelszóadatbázis rám vonatkozó feljegyzéseit ne tudja magát az én nevemben azonosítani***. Ez csak úgy oldható meg, ha a ***jelszavakat titkosítva tároljuk***. Vegyük észre, hogy az sem jó megoldás, ha a titkosító függvény kulcsot használ, hiszen a titkosító kulcsot is tárolni kell valahol. A jelszavak titkosításához tehát egy kulcs nélküli egyirányú függvényt célszerű használni, amelyet $h$-val fogunk jelölni.

Most már fel tudjuk vázolni a jelszavas azonosítás folyamatát. Három fázisa van; bemutatkozás (regisztráció), működés és megszüntetés.

- **Bemutatkozás.** A felhasználó közvetlenül vagy közvetítő, pl. adminisztrátor, útján közli a számítógéppelaz azonosításhoz szükséges adatokat többek között a jelszavát is, amelyet $p$-vel

jelölök. A számítógép a felhasználónak nyit egy rekordot a jelszó állományában, ahova elmenti az adatokat. A jelszót azonban titkosított formában tárolja. Ez annyit jelent, hogy nem a *p*, hanem a *h(p)* érték szerepel az adatbázisban.

- **Működés.** Amikor a felhasználó igénybe akarja venni a számítógép szolgáltatásait, akkor „bejelentkezik oda", azaz megadja a felhasználói nevét és jelszavát. Utóbbit jelöljük ismét *p*-vel. A számítógép megnézi, hogy a jelszó adatbázisában szerepel-e ilyen nevű felhasználó. Ha igen, akkor megnézi, hogy a megadott jelszó titkosított változata *h(p)* tartozik-e a bejelentkezni akaró felhasználóhoz. Ha igen, akkor beengedi a rendszerbe. Ha az előző két kérdés bármelyikére nemleges a válasz, akkor a számítógép elutasítja a kérelmet.

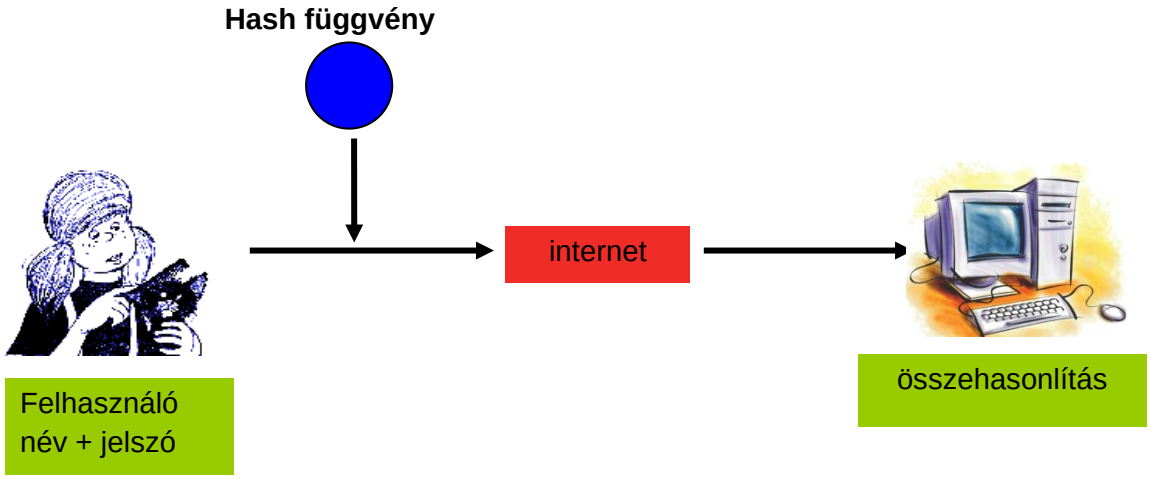
- **Megszüntetés.** A felhasználó adatait törlik a jelszóadatbázisból.

Az általános sémának nagyon sok változata van. Még mindig felbukkannak olyan azonosító programok, amelyek a jelszavakat titkosítás nélkül tárolják. Nem szabad ilyeneket használni! A *h(p)* érték kiszámítása csak néhány pillanatig tart, sokkal biztonságosabb azonban ezt mint *p*-t tárolni.

A regisztráció során az első jelszót adhatja a számítógép, amelyet később kicserélünk, esetleg ki kell cserélnünk, az érvényes jelszó azonban minden esetben titkosítva szerepel a jelszóadatbázisban.

A hozzáférési jogosultság ideiglenes szüneteltetése például munkaidőn kívül és szabadságon nem az azonosítás, hanem a jogosultságkezelés feladata.

A fent leírt működési mechanizmus biztonságos akkor, amikor a felhasználó közvetlen összeköttetésben áll a számítógéppel. Abban az esetben azonban, amikor az interneten keresztül vagy egy nem jól karbantartott internet kávézóból jelentkezünk be az esetleg több ezer kilométerre található számítógépbe, akkor újabb probléma jelentkezik. Ha a jelszó titkosítása a számítógépen történik, akkor a bejelentkezés helyétől a célállomásig elvileg bárki elolvashatja és társíthatja a felhasználó névvel. Biztonságosnak ezért csak az a jelszavas azonosító rendszer tekinthető, amelyik a jelszót a felhasználó oldalán titkosítja. Ilyet mutat az 1. ábra.



1. ábra

*Tesztkérdések:*

T38. A hozzáférési _____ ideiglenes _____ munkaidőn kívül és szabadságon nem az _____, hanem a _____ feladata. Egészítse ki az előző hiányos mondatot az alábbi szavak közül azokkal, amelyek oda illenek: jogosultság, idő, szüneteltetése, gyakorlása, azonosítás, jelszavak, rendszergazda, jogosultságkezelés.

T39. Biztonságos azonosító rendszerben a jelszavakat a

a) szerver oldalon kódolják
b) kliens oldalon kódolják
c) nem kódolják

T40. Tegye helyes sorrendbe a jelszavas azonosítás működési fázisait leíró szavakat: 1: Működés, 2: Megszűntetés, 3: Regisztráció.

## 3.3 Szótáras támadás a jelszavak ellen; gyenge és erős jelszavak

A jelszó titkosításával, majd a titkosítás kliens oldali telepítésével az azonosítás technológiája megfelel a szokásos biztonsági követelményeknek. A mechanizmus leggyengébb láncszeme a felhasználó lett. Bár az elmúlt 20 évben a számítógéppel támogatott munkahelyek, az internet és az e-alkalmazások elterjedésével lényegesen javult a helyzet, sokakban máig sem tudatosult a számítógépes azonosítás fontossága. Egy név alatt többen használják ugyanazt az alkalmazást, a felhasználó nevet és a jelszót a monitor keretére írják vagy ragasztják, stb. Sok olyan felhasználó van, aki tudatos számítógép-használó, de mégsem használ biztonságos jelszót.

De milyen is a biztonságos jelszó és miért olyan? Az azonosítás során a jelszavakat egy $h$ egyirányú függvénnyel titkosítva tároljuk. Ha a $h$-t alaposan leteszteljük, akkor bízhatunk benne, hogy a tárolás biztonságos lesz, azaz a titkosított értékből - $h(p)$ - nagyon nehéz $p$-t kitalálni. Hasonló helyzettel állunk szemben, mint amikor a nyomtatott telefonkönyvben a telefonszámhoz az előfizetőt kell megtalálni. Az utóbbit nehéz feladatnak ismerjük, ezért az előzőt is nehéznek gondoljuk. Sajnos ez csak ***tudatos jelszóválasztás*** mellett van így. A jelszavak egy részét ugyanis a ***szótáras támadás***sal meg lehet fejteni.

A szótáras támadás két dolgot használ ki:

a $h$ egyirányú függvény algoritmusa nyilvános és

a felhasználók olyan jelszavakat választanak, amelyeket könnyen megjegyeznek.

A hacker először összeállít egy bőséges listát a gyakran használt szavakból; egy lexikont és/vagy szótárat kiegészít családi- és keresztnevekkel, állatok, növények, gyakran használt tárgyak, zenekarok, celebek, stb. neveivel. Tovább bővítik a listát a szavak egyszerű transzformáltjaival,

például a szavakban az első és második betűt felcserélik, stb. A lista bővítésének csak a készítő fantáziája, a munkára fordítható idő és a rendelkezésre álló tárkapacitás szab határt. Ha a hacker hozzájut egy jelszóadatbázishoz, akkor megkeresi, hogy az alkalmazás milyen egyirányú függvénnyel titkosítja a jelszavakat. Jelöljük ezt *h*-val. Ezek után a hacker minden szóra alkalmazza a *h* függvényt, előállítva így a *(p,h(p))* párokból álló szótárat. A jelszóadatbázisban a titkosított jelszavakat tartalmazó rekordok *h(q)* alakúak, ahol *q* az ismeretlen jelszó. Ha a hacker a szótárában talál olyan beírást, amelynek a második tagja megegyezik *h(q)*-val, akkor megfejtette a titkot.

A szótáras támadás jól ismert, alkalmazzák arra is, hogy a felhasználók gyenge jelszavait, amelyek szótáras támadással megfejthetőek, kiszűrjék. Az angol abécében 26 betű, a magyarban, a többes betűket nem számítva, 35 betű. A kis- és nagybetűket a számítógép eltérő módon tárolja, ha tehát még ezeket is figyelembe vesszük, akkor 52 betűvel számolhatunk angol, 70-nel magyar szövegek esetén. A 0-9 számjegyeket és írásjeleket, mint +, !, §, %, /, =, (, ), \, |, ×, $,<, >, #, &, @, {, }, ?, ;, :, * is megengedve további tízzel, illetve 23-mal nő a különböző karakterek száma. Az 1. táblázat bemutatja, hogy ezekből hányféle öt - nyolc betűből álló szót lehet alkotni. Természetesen ezeknek a döntő többsége értelmetlen. A táblázat adatait könnyű kiszámítani, hiszen jól tudjuk, hogy *n* betűből pontosan $n^k$ darab különböző *k* hosszúságú karaktersorozat képezhető.

| | Kisbetű | Kis- és nagybetű | Kis- és nagybetű és szám | Kis- és nagybetű +szám + különleges karakter |
|---|---|---|---|---|
| Angol/5 | $1{,}2\ 10^7$ | $3{,}8\ 10^8$ | $9{,}16\ 10^8$ | $4{,}44\ 10^9$ |
| Angol/6 | $3{,}1\ 10^8$ | $1{,}98\ 10^{10}$ | $5{,}68\ 10^{10}$ | $3{,}77\ 10^{11}$ |
| Angol/7 | $8{,}032\ 10^9$ | $1{,}03\ 10^{12}$ | $3{,}52\ 10^{12}$ | $3{,}21\ 10^{13}$ |
| Angol/8 | $2{,}09\ 10^{11}$ | $5{,}35\ 10^{13}$ | $2{,}18\ 10^{14}$ | $2{,}72\ 10^{15}$ |
| Magyar/5 | $5{,}25\ 10^7$ | $1{,}68\ 10^9$ | $3{,}28\ 10^9$ | $1{,}16\ 10^{10}$ |
| Magyar/6 | $1{,}84\ 10^9$ | $1{,}18\ 10^{11}$ | $2{,}62\ 10^{11}$ | $1{,}19\ 10^{12}$ |
| Magyar/7 | $6{,}44\ 10^{10}$ | $8{,}24\ 10^{12}$ | $2{,}1\ 10^{13}$ | $1{,}23\ 10^{14}$ |
| Magyar/8 | $2{,}25\ 10^{12}$ | $5{,}76\ 10^{14}$ | $1{,}68\ 10^{15}$ | $1{,}27\ 10^{16}$ |

1. táblázat

A táblázatból jól látható, hogy a szavak száma az abécé nagyságától és a szavak hosszától függ. Az angol abécé kis- és nagybetűiből és a számjegyekből nagyjából $10^9$ darab 5 karakter hosszúságú szó képezhető. Ez nagyjából 1 GB adatmennyiség, amelyet manapság néhány másodperc alatt fel lehet dolgozni. Ilyen jelszavakat tehát könnyedén fel lehet törni. Ezzel szemben a magyar abécé kis- és nagybetűivel, számjegyekkel és különleges karakterekkel leírható 8 betűs szavak már csaknem *13000* TB nagyságú szótárat eredményeznek, amelynek kezelése a ma rendelkezésünkre álló számítási kapacitás határán van.

Ezek után érthető, hogy miért tanácsolják, hogy a jelszavak tartalmazzanak kis- és nagybetűt, valamint különleges karaktereket és legyenek legalább 6, de inkább 8 karakter hosszúságúak.

### Tesztkérdések:

T41. A szótáras támadás
   a) lehetőséget ad a gyenge jelszavak megszerzésére,
   b) a pszichológiai megtévesztés egyik eszköze,
   c) megvédi a jelszavakat a hackerektől.

T42. A jelszóban
   a) legyen különleges karakter, mert így biztonságosabb,
   b) ne legyen különleges karakter, mert ezzel gyengítjük,
   c) ne legyen különleges karakter, mert akkor nehezebb megjegyezni.

T43. Hol tárolja a számítógépes jelszavait?
   a) A monitor keretére írva, mert akkor nem felejtem el.
   b) Megjegyzem őket.
   c) Beírom a mobiltelefonomba.
   d) Elmondom a partneremnek, akitől szükség esetén megkérdezhetem.
   e) Gyorsan elfelejtem őket.

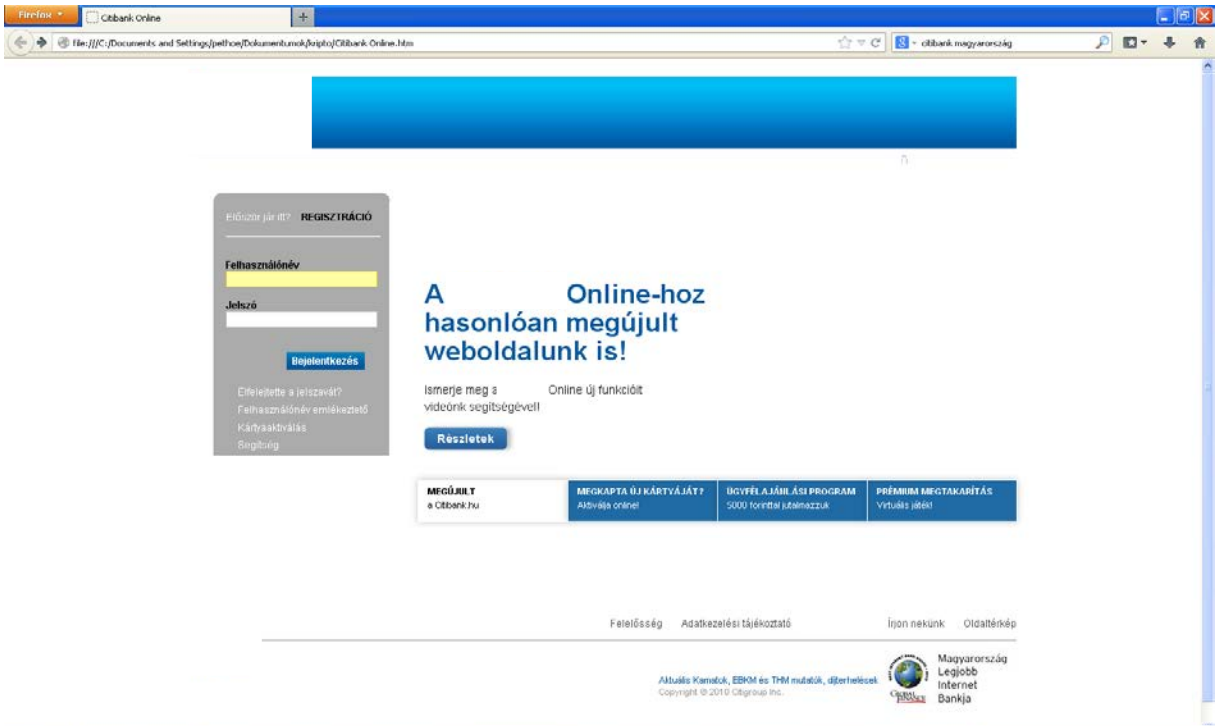
## 3.4 Adathalászat

Az adathalászat, a jelszavak csalárd eszközökkel való megszerzése, nem a kriptográfia, hanem a social engineering, azaz a *pszichológiai manipuláció* területéhez tartozik. Olyan veszélyes technikát jelent, amely ellen való védekezésre minden adódó alkalmat meg kell ragadni. ***A pszichológiai manipuláció olyan megtévesztő módszerek megismerésével, leírásával és fejlesztésével foglalkozik, amelyek az emberek kíváncsiságát, jóhiszeműségét és segítőkészségét és egyéb tulajdonságait használják ki***, a hackerek elsőszámú eszköze. Célja a társadalmi együttéléshez nélkülözhetetlen pozitív tulajdonságainkat kihasználva számunkra is káros következményekkel járó cselekedetek tudatos megtervezése. Kevin D. Mitnick könyve[9] tanulságos példákat tartalmaz.

***Az adathalászat célja valamely szolgáltatás igénybevételére feljogosító jelszavak megszerzése.*** Leggyakrabban a közvetlenül hasznosítható banki jelszavak állnak a célkeresztben, de az elektronikus postaládák jelszavai is kedveltek. Példaként bemutatjuk banki jelszavak kihalászásának technikáját és az eredményes halászat következményét. A 2. ábrán bemutatott

[9]Kevin D. Mitnick és William L. Simon, A legendás hacker: A behatolás művészete, perfact kiadó, 2006.

hamis banki honlapra mutató linket a szerző kapta 2013-ban. Nem volt soha ügyfele ennek a banknak.



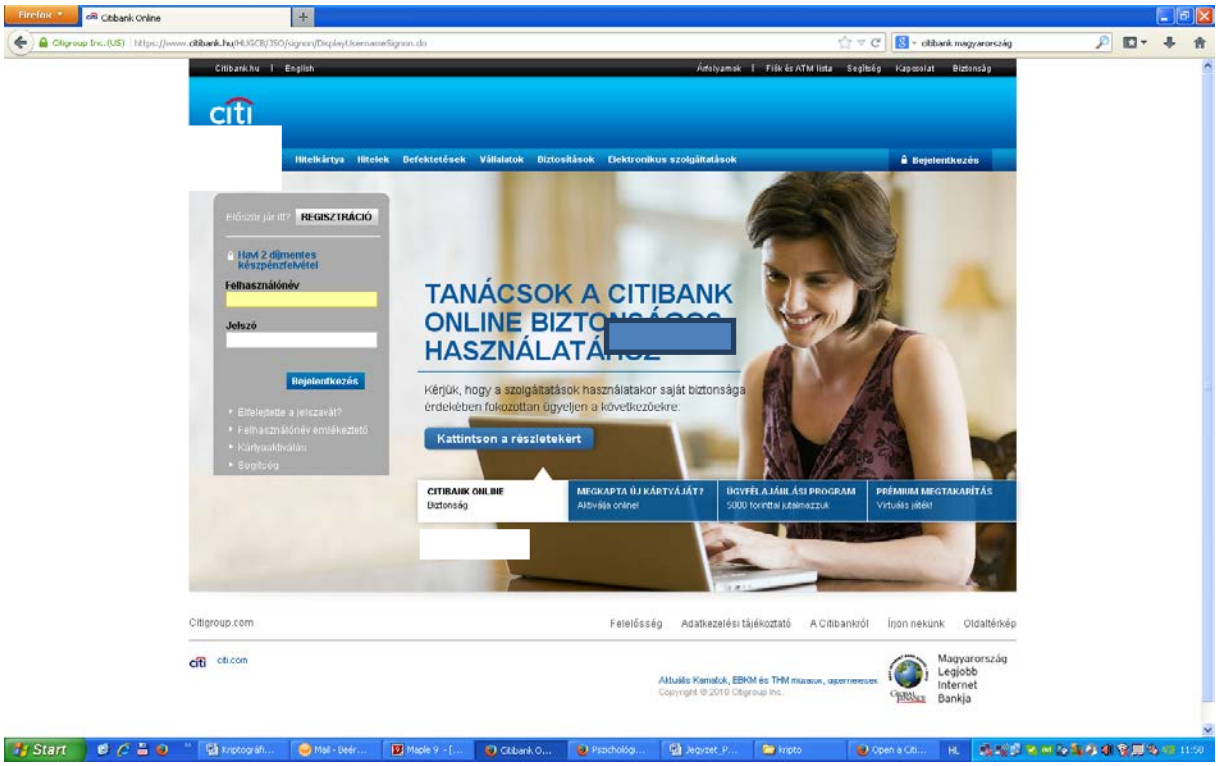

2. ábra

Az adathalász a webről összegyűjt e-mail címeket. Ehhez elegendő egy egyszerű kereső robotot[10] írnia, amelyik végigkutatja a webet. Amíg a robot a webet vizslatja, az adathalász elkészíti az áldozatul kiszemelt bank honlapjának a hasonmását. A hamisított honlap címét egy levélbe másolja, amelyben a bank nevében, például adatfrissítésre hivatkozva kéri, hogy a címzett lépjen be a bankjába. Ezután a leveleket elküldi minden összegyűjtött e-mail címre. A sok címzett között biztosan van néhány banki ügyfél is.

Az eredeti és a hamisított honlap két fontos dologban különbözik; az egyiket a felhasználó is láthatja, a másiknak csak a kellemetlen következményét érzékeli.

Az eredeti banki honlapon, ld. 3. ábra, a böngésző sor bal sarkában jól látható egy kis lakat, amely azt jelzi, hogy a bejelentkezés, majd az utána történő információcsere lehallgathatatlan csatornán keresztül történik. Ezt a lakatra kattintva előugró tanúsítvány bizonyítja. A megoldás technológiájáról a 4.4 és az 5. fejezetekben lesz szó.

[10] A robot itt nem egy fizikai, hanem szoftveres eszközt jelent. Tulajdonképpen egy program, amelynek a feladata weblapok végignézése valamilyen meghatározott célból. Esetünkben a weblapokon található e-mail címek összegyűjtése a feladata.




Az eredeti és a hamisított honlap közötti másik különbség az, hogy a felhasználó név és a jelszó nem ugyanoda megy. A banki honlap esetén a bank szerverébe kerül, amelyik azokat a 3.2. fejezetben leírtak szerint dolgozza fel. Ezzel szemben a hamisított honlapról az adatok a hacker számítógépére kerülnek, aki rögtön kezdeményezi a belépési eljárást a bank igazi honlapján. Ha szerencséje van, akkor a részedett felhasználó számlájáról valamilyen nagy összegű tranzakciót hajt végre. Az adathalászat egyéb vonatkozásaival terjedelmi okok miatt nem tudunk foglalkozni. ***Olyan e-mailre azonban, amelyben az azonosítóik megadását kérik soha se válaszoljanak, haladéktalanul töröljék azokat!***

*Tesztkérdések:*

T44. A pszichológiai manipuláció olyan _____ módszerek megismerésével, leírásával és fejlesztésével foglalkozik, amelyek az emberek _____, jóhiszeműségét és _____ használják ki. Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: kedves, előnyös, megtévesztő, hiszékenységét, szabadságát, lustaságát, segítőkészségét.

T45. Az adathalászat

a) Csak tengervízben történhet,
b) egy módszer a jelszavak megszerzésére,
c) a jogosultságkezelés egyik gyakran használt módszere.

T46. Megadjuk-e jelszavunkat, ha e-mailben kérik?

a) Semmi esetre sem.
b) Igen, mert ezzel segítjük az adminisztrátor munkáját.
c) Teljesen mindegy.

# 4 Az aszimmetrikus titkosítás alapjai

Az aszimmetrikus, más szóval nyilvános kulcsú kriptorendszerek fő ismérve, hogy a titkosító kulcs és a visszafejtő kulcs nemcsak hogy különbözik egymástól, de emberi léptékű időn belül nem is számíthatók ki egymásból, legalábbis egy bizonyos titok ismerete nélkül. A fenti tulajdonság így lehetővé teszi, hogy a kriptorendszer minden résztvevője készítsen, vagy igényeljen magának egy személyre szóló kulcspárt, és ezek közül a titkosító kulcsot nyilvánosságra hozza. Amennyiben valaki számára titkos üzenetet kíván küldeni, akkor ezt a nyilvánosságra hozott titkos kulcsa segítségével titkosítva megteheti, és ezt csak a címzett, az általa titokban tartott visszafejtő kulcsával képes olvasható szöveggé konvertálni.

A nyilvános kulcsú kriptorendszer lényegének szemléltetése„hétköznapi" példával:

> *Képzeljük el, hogy Aladár szeretne úgy üzeneteket kapni barátaitól, hogy azt családtagjai és kedves szomszédai (különös tekintettel Máris szomszédra) ezekhez az üzenetekhez ne férhessenek hozzá. Készít hát egy olyan széfet, melynek az ajtaja az utcára néz, és amelynek számzára kombinációját csak Ő ismeri. Aztán ezt a széfet nyitva hagyja. Aki üzenetet akar küldeni neki, az ezt elhelyezi a széfben, majd jó alaposan rázárja a széf ajtaját. Ezek után már csak Aladár férhet hozzá az üzenethez. Megváltoztatni vagy eltávolítani azt már az üzenet küldője sem tudja.*

Egy nyilvános kulcsú kriptorendszer lényegében a fenti alapötlet matematikai megvalósítása.

*Történeti elemek*:

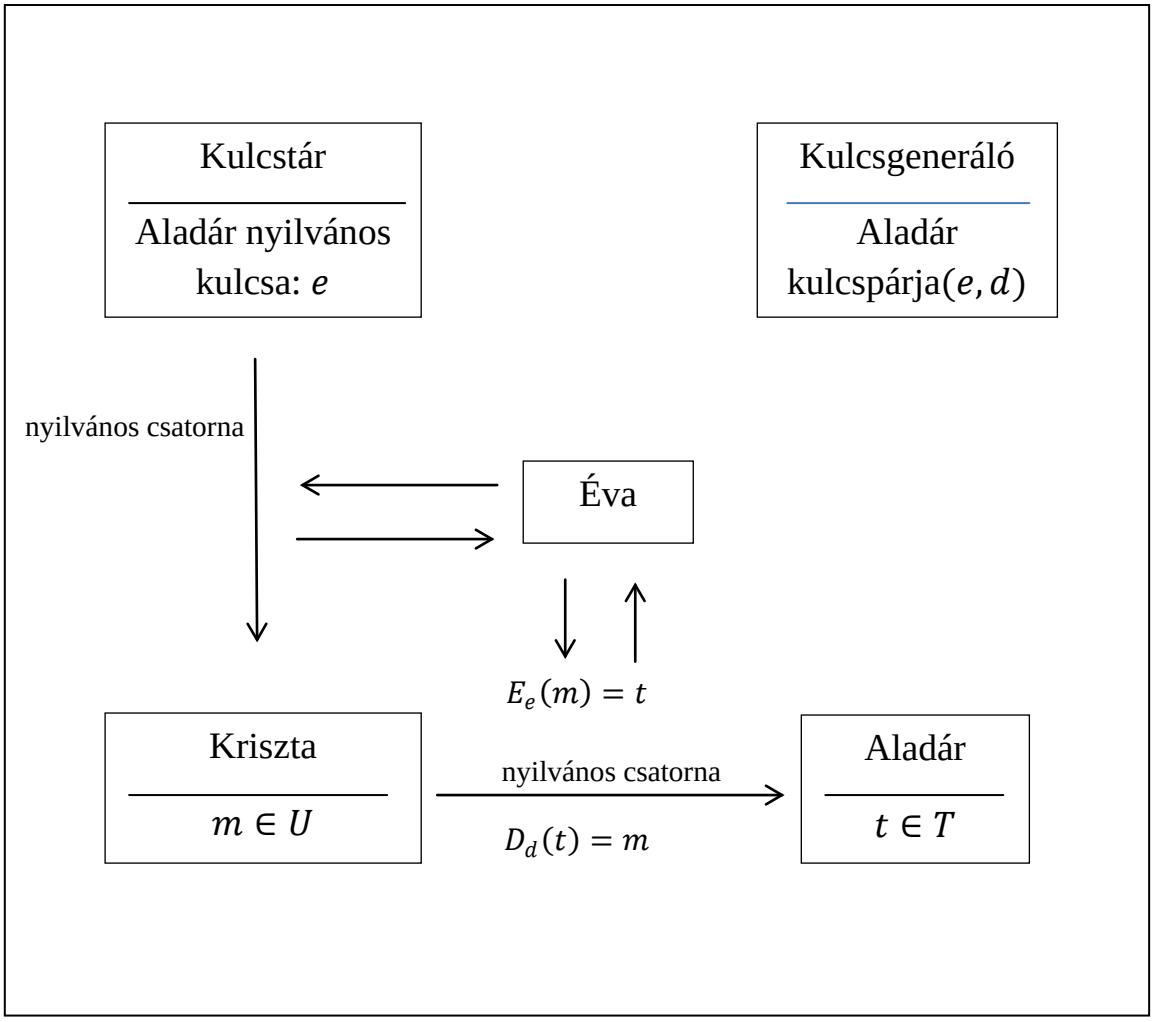
- A nyilvános kulcsú titkosítás alapötletét először Whittfield Diffie és Martin E. Hellman publikálta 1976-ban
- 1977-ben Ronald Rivest, Adi Shamir és Leonard Adleman publikálta az első nyilvános kulcsú kriptorendszert, amelyet a vezetékneveik kezdőbetűit összeolvasva RSA névre kereszteltek el.
- A dolog pikantériája, hogy 1997-ben nyilvánosságra hozták, hogy a Brit Kormányzati Kommunikációs Főhadiszállás (British Government Communications Headquoters) egyik csoportjának tagjaként James Ellis egy cikkében már 1970-ben felvetette a Diffie és Hellman dolgozatában szereplő ötletet, és Clifford Cocks 1973-ban írt cikkében már javaslatot tesz lényegében ugyanazon kriptorendszer használatára, mint később Rivest, Shamir és Adleman. Ezek a dolgozatok azonban 1997-ig nem kerültek nyilvános publikálásra, így nyilván szerzőik sem kapták meg érte a megérdemelt tudományos elismerést. Ezt természetesen tekinthetjük a titkos kormányzati munka egyik árnyoldalának.

*A nyilvános kulcsú kriptorendszer matematikai modellje*:

Egy nyilvános kulcsú kriptorendszer alatt egy $(U, K, T, \{E_e\}, \{D_d\})$ ötöst értünk, mely egyrészt teljesíti a kriptorendszerre vonatkozó korábbi kitételeket, továbbá

- az $E_e$ titkosító függvények olyan egyirányú csapóajtó függvények, melyeknek az inverze emberi léptékű idő alatt kizárólag a $d$ kulcs ismeretében számítható ki.
- az $(e, d)$ kulcspárok közül az $e$ minden esetben nyilvános információ, amihez bárki hozzájuthat.
- az $e$ ismeretében a $d$ kiszámítása lényegében lehetetlen, legalábbis valamely további, titkos ismeret birtoklása nélkül.

*A nyilvános kulcsú kriptorendszer diagrammja:*



## 4.1 Az RSA kriptorendszer

Az RSA kriptorendszer azon az ötleten alapszik, hogy két nagy prímszám szorzatát (ezen prímszámok előzetes ismerete nélkül) nagyon nehéz (azaz nagyon sok időbe telik) faktorizálni.

### Az RSA kriptorendszer leírása:

*Kulcsgenerálás:*

1. Aladár generál két nagyjából azonos méretű $p \neq q$ prímszámot
2. Kiszámítja az $n = p \cdot q$ és a $\varphi(n) = (p-1)(q-1)$ értékeket
3. Rögzít egy véletlen $e \in \mathbb{N}, 1 < e < \varphi(n)$ számot, melyre $\gcd(e, \varphi(n)) = 1$
4. A kiterjesztett Euklideszi algoritmus segítségével kiszámítja azt a $d \in \mathbb{N}$ számot, melyre $1 < d < \varphi(n)$ és
$$ed \equiv 1 \pmod{\varphi(n)}$$
5. Aladár nyilvánosságra hozza az $(n, e)$ párt, és titokban tartja a $d, p, q$ és $\varphi(n)$számokat.

*Elnevezések:*

- $(n, e)$ Aladár nyilvános kulcsa
- $d$ Aladár titkos kulcsa (más néven privát kulcsa)
- e Aladár (RSA) titkosító kitevője
- $d$ Aladár (RSA) visszafejtő kitevője

*Titkosító fázis:*

Az egyszerűség kedvéért tegyük fel, hogy az $m \in U$ nyílt szöveg egy $m < n$ természetes szám formájában van kódolva, továbbá $U = T = \{0,1,2, \dots, n-1\}$ és $\gcd(m, n) = 1$.

1. Kriszta megkeresi Aladár$(n, e)$ nyilvános kulcsát az adatbázisból
2. Kiszámolja a $t \equiv m^e \pmod{n}$, $1 \leq t < n$ értéket, így titkosítva az $m$ üzenetblokkot
3. Elküldi a $t \in T$ értéket Aladárnak.

*Visszafejtő fázis:*

Amikor Aladár megkapja a $t$ értéket, akkor kiszámítja az $m \equiv t^d \pmod{n}$, $1 \leq m < n$ értéket, így megkapva az eredeti üzenetet.

### Paraméterválasztás az RSA kriptorendszerhez:

- Az $n$ modulus mérete olyan nagy kell legyen, hogy annak faktorizációja lényegében lehetetlen legyen. A számítástudomány mai állása szerint egy jól megválasztott 1024 bites

(azaz $2^{1024}$ körüli) modulus biztonságosnak tekinthető, de 4096-bites modulusok várhatóan még évtizedekig megfelelőek lesznek

- A $p$ és $q$ prímek tehát úgy választandók, hogy a méretük legalább 512 bites legyen. Ezen túlmenően azonban ahhoz, hogy az $n$ faktorizációja nehéz legyen, még figyelnünk kell arra, hogy
  - $p$ és $q$ mérete nagyságrendileg azonos legyen
  - $p$ és $q$ különbségének a mérete is $p$ és $q$ nagyságrendjébe essen, ugyanis ha $p-q$ kicsi, akkor létezik hatékony faktorizációs módszer $n$ faktorizálására
  - a $\varphi(n)=(p-1)(q-1)$ számnak legyen egy nagy prímfaktora
  - a $p-1$ és $q-1$ legnagyobb közös osztója legyen nagy
- bár az $e$ kitevő elvileg szabadon választható az $1<e<\varphi(n)$ és $\gcd(e,\varphi(n))=1$ feltételek mellett, el kell kerülni néhány nagyon rossz választást, mint például
  - $e=\frac{\varphi(n)}{2}+1$
  - $e$ kis értékei a titkosítás sebességének növelése szempontjából jó választásnak tűnhetnek, de ezt is mindenképpen érdemes elkerülni, mert ha ugyanazt az üzenetet több személynek is el akarjuk küldeni, kis titkosító kitevő mellett a nyílt szöveg visszakereshető

### Az RSA sejtés:

*Az RSA kriptoanalízise ugyanolyan nehéz feladat, mint az $n$ modulus faktorizációja.*

Ez a sejtés garantálja az RSA biztonságát.

### Blokkok kialakítása a nyílt szövegben

Felmerül a kérdés, hogy hogyan tudjuk garantálni, hogy az eredeti üzenet egy $m<n$ numerikus értékkel legyen ábrázolva (kódolva). Nyilván, ha az üzenet rövid, ez nem jelent problémát, amint azt az 1.4 Fejezetben már tárgyaltuk.Amennyiben viszont hosszú üzenetet kell titkosítanunk, akkor a következőek szerint járunk el:

- Tegyük fel, hogy a nyílt szöveg eredetileg egy $N$ elemű abécé feletti nyelven íródott.
- Legyen $l$ az az egyetlen természetes szám, melyre $N^l \le n < N^{l+1}$.
- A szöveget bontsuk fel $l$-betűs blokkokra.
- Minden blokknak megfelel egy $l$ számjegyű szám az $N$ alapú számrendszerben, feltéve, hogy az utolsó blokkot szükség esetén nullákkal egészítjük ki (jobbról).
- Mivel $N^l \le n \le N^{l+1}$, így minden nyílt szöveg megfelel a $\{0,1,2,\dots,n-1\}$ halmaz egy elemének, így a titkosított szöveg is a $\{0,1,2,\dots,n-1\}$ halmaz egy eleme lesz, azaz $n<N^{l+1}$ miatt egy $l+1$ jegyű számmal ábrázolható az N alapú számrendszerben.

- Ezek után az egyes blokkokat külön-külön titkosítjuk az RSA segítségével, és az így kapott titkosított szövegek egymás után fűzésével kapjuk az eredeti üzenet titkosítását.

**Fontosabb támadási lehetőségek az RSA ellen[11]:**

- Az $n$ faktorizációja
- Implementációs hiányosságokat kihasználó támadások
  - Közös modulus protokoll-hiba.
  - Időméréses támadás.
  - Áram-kriptoanalízis.
- Támadások a kitevő ellen

## 4.2 A diszkrét logaritmus problémán alapuló kriptorendszerek

### 4.2.1 A diszkrét logaritmus probléma (DLP):

***Definíció.*** Legyen $m$ egy pozitív egész szám. Egy $g$ egész számot primitív gyöknek nevezünk modulo $m$ ha minden $b \in \{1,2,\dots,m-1\}$ szám esetén, melyre $(b,m)=1$ létezik olyan $k$ pozitív egész szám, melyre $g^k \equiv b \pmod m$.

***Tétel.*** Akkor, és csakis akkor létezik primitív gyök modulo $m$, ha $m=2, m=4, m=p\text{^}k$ vagy $m=2p\text{^}k$, ahol $p$ palamely pozitív prímszám.

***Definíció.*** Legyen $p$ egy pozitív prímszám, $b$ egy egész szám, melyre $(b,p)=1$, és legyen $g$ egy primitív gyök modulo $p$. Azt a legkisebb pozitív egész $k$ számot, melyre teljesül, hogy $g^k \equiv b \pmod p$ a $b$ szám $g$ alapú diszkrét logaritmusának nevezzük modulo $p$.

***Miért lehet a diszkrét logaritmus problémát a nyilvános kulcsú kriptográfiában használni?***
Miközben a modulo $p$ történő hatványozás nagy $p$ értékek esetén is gyorsan kiszámítható, addig ennek megfordítása, a diszkrét logaritmus kiszámítása nagyon időigényes feladat.

### 4.2.2 Az ElGamal kriptorendszer

Kriszta szeretne egy $m \in \{0,1,2,\dots,p-1\}$ üzenetet elküldeni Aladárnak

***Kulcsgenerálás:***

1. Aladár választ egy nagy $p$ véletlen prímet (amely esetén a DLP megoldása „szinte lehetetlen"), és egy $\alpha$ primitív gyököt modulo $p$.
2. Aladár ezután választ egy véletlen $2 \le a < p-1$ természetes számot, és kiszámolja a $\beta = \alpha^a \pmod p$ értéket.

[11] Itt csak megemlítjük a legfontosabb támadási lehetőségeket. Az érdeklődő olvasót Mollin [1] könyvének tanulmányozására bíztatjuk

3. *Aladár nyilvános kulcsa*: $(p, \alpha, \beta)$
   *Aladár titkos kulcsa*: $a$

*Titkosító fázis:*

1. Kriszta megszerzi Aladár nyilvános kulcsát: $(p, \alpha, \beta)$.
2. Kriszta ezután választ egy véletlen $2 \leq b < p - 1$ természetes számot.
3. Kriszta kiszámolja az $\alpha^b \pmod p$ és $m\alpha^{ab} = m\beta^b \pmod p$ értékeket.
4. Kriszta elküldi a $t = (\alpha^b \pmod p, m\alpha^{ab} \pmod p))$ titkosított üzenetet Aladárnak.

*Visszafejtő fázis:*

1. Aladár felhasználva a saját titkos kulcsát kiszámítja az
$$(\alpha^b)^{-a} \equiv (\alpha^b)^{p-1-a} \pmod p$$
értéket.
2. Aladár ezután kiszámítja az $(\alpha^b)^{-a} m\alpha^{ab} \equiv m \pmod p$ értéket, azaz visszanyeri $m$ értékét.

*Megjegyzések az ElGamal kriptorendszerrel kapcsolatosan:*

- Az ElGamal kriptorendszer egyik fő előnye, hogy nem determinisztikus, azaz a $b$ (Kriszta által választott) véletlen szám miatt, egyazon nyílt szöveg többszöri elküldése esetén, a titkosított szöveg más és más lesz.
- Látható, hogy az ElGamal kriptorendszer esetén a titkosított szöveg körülbelül kétszer olyan hosszú, mint a nyílt szöveg. Ezt a jelenséget hívjuk „üzenet expanziónak", és ez az ElGamal kriptorendszer egyik fő hátránya.
- Ha az üzenet numerikus kódja $m > p$-1 akkor az RSA-nál ismertetett módon az eredeti nyílt szöveget blokkokra kell bontani.

### 4.2.3 A Massey-Omura kriptorendszer

Legyen $p$ egy nagy prímszám. Kriszta szeretne egy $m \in \{0,1,2, \dots, p-1\}$ üzenetet elküldeni Aladárnak. Ehhez Kriszta és Aladár az alábbi lépéseket hajtják végre:

1. Kriszta és Aladár egymástól függetlenül választanak egy-egy $2 \leq e_A, e_B < p - 1$ egész számot, melyekre $(e_A, p-1) = 1$ és $(e_B, p-1) = 1$. Nyilván az $e_A$ értéket csak Kriszta, az $e_B$ értéket csak Aladár ismeri.
2. Kriszta és Aladár kiszámolja az $e_A$ és $e_B$ inverzét modulo $p$ a kiterjesztett Eulideszi algoritmussal, azaz meghatározzák a $d_A \equiv e_A^{-1} \pmod p$és $d_B \equiv e_B^{-1} \pmod p$számokat.
3. Kriszta elküldi az $m^{e_A} \pmod p$ értéket Aladárnak.
4. Aladár visszaküldi az $m^{e_A e_B} \equiv (m^{e_A})^{e_B} \pmod p$ értéket Krisztának.

5. Kriszta elküldi az $m^{e_A e_B d_A} \equiv (m^{e_A e_B})^{d_A} \equiv (m^{e_A d_A})^{e_B} \equiv m^{e_B} \pmod{p}$ értéket Aladárnak.
6. Aladár ebből kiszámítja az $(m^{e_B})^{d_B} \equiv m \pmod{p}$ értéket, azaz megkapja az eredeti üzenetet.

*Megjegyzések a Massey-Omura kriptorendszerrel kapcsolatosan:*

- A Massey-Omura kriptorendszer nem szigorúan vett nyilvános kulcsú kriptorendszer, hiszen nincs se nyilvános kulcs, se megosztott titkos kulcs.
- Az első lépés után Aladár nem tudja $m^{e_A} \pmod{p}$ birtokában meghatározni az m üzenetet, mert nem ismeri a $d_A$ kitevőt.
- Aladár az $m^{e_A} \pmod{p}$ birtokában nem tudja az $e_A$ kitevőt sem meghatározni (sőt ezt később $m$ birtokában sem képes megtenni), hacsak nem tudja megoldani a diszkrét logaritmus problémát modulo $p$.
- Kriszta$m^{e_B} \pmod{p}$ és $m$ birtokában szintén nem képes meghatározni az $e_B$ kitevőt, hacsak nem tudja megoldani a diszkrét logaritmus problémát modulo p.
- A fentiektől függetlenül természetesen azért a Massey-Omura kriptorendszer használata során minden alkalommal új véletlen kitevőket kell használni.
- Megjegyezzük, hogy a Massey-Omura kriptorendszer alapötlete minden olyan titkosító és visszafejtő függvénycsalád esetén alkalmazható, ahol a családba tartozó függvények alkalmazásának sorrendje felcserélhető, azaz a sorrend megváltoztatása a végeredményt nem befolyásolja. Ennek megfelelően a modulo p vett diszkrét logaritmus probléma helyett választhatunk egy általánosabb struktúra feletti diszkrét logaritmus problémát.
- A Massey-Omura kriptorendszer egyik hátránya, hogy három üzenetküldésre van szükség.
- Egy másik, ennél sokkal súlyosabb probléma, hogy ez a kriptorendszer teljesen védtelen a "man in the middle" támadással szemben, azaz megeshet, hogy aKriszta által küldött üzeneteteket Mallory elfogja, és úgy tesz, mintha ő lenne Aladár így valójában Kriszta tudtán kívül Aladár helyett Malloryval levelezik, és erről Aladár még csak nem is tud. Ezt a támadó szándékai szerint azzal is megtoldhatja, hogy őKrisztát megszemélyesítve eljuttatja az üzenetet Aladárnak, így Kriszta és Aladár azt hiszik, hogy minden rendben ment, míg Mallory észrevétlenül jutott hozzá a titkukhoz.

*Tesztkérdések:*

T47. Az alábbi estek közül melyekben létezik primitív gyök modulo $m$?
   a) m=17
   b) m=12
   c) m=27
   d) m=8

T48. Mennyi az 5 szám 2 alapú diszkrét logaritmusa modulo 11?
   a) 7
   b) 2
   c) 4
   d) 5

T49. Melyik évben publikálta Diffie és Hellman a nyilvános kulcsú titkosítás alapötletét?
   a) 1972
   b) 1976
   c) 1980
   d) 2000

T50. Melyik volt az első publikált nyilvános kulcsú kriptorendszer?
   a) DES
   b) AES
   c) RSA
   d) ElGamal

T51. Honnan származik az RSA kriptorendszer elnevezése?
   a) A megalkotók nevének kezdőbetűiből.
   b) A nyilvános kulcsú kriptorendszer angol nevének rövidítése.
   c) Az USA szabványügyi hivatalának nevéből.
   d) A jogtulajdonos cég nevéből.

T52. Melyik probléma megoldásának nehéz voltán alapszik az RSA kriptorendszer biztonsága?
   a) A hátizsák probléma.
   b) A diszkrét logaritmus probléma.
   c) A gyakoriságanalízis problémája.
   d) A faktorizáció problémája.

T53. Melyik probléma megoldásának nehéz voltán alapszik az ElGamal kriptorendszer biztonsága?
   a) A hátizsák probléma.
   b) A diszkrét logaritmus probléma.
   c) A gyakoriságanalízis problémája.
   d) A faktorizáció problémája.

T54. Az alábbiak közül melyik kriptorendszer biztonsága alapszik a faktorizáció problémájának nehézségén?

a) AES
b) ElGamal
c) RSA
d) DES

T55. Az alábbiak közül melyik kriptorendszerek biztonsága alapszik a diszkrét logaritmus probléma nehézségén?
a) AES
b) ElGamal
c) RSA
d) DES

T56. Az alábbiak közül melyik kriptorendszer hátrányos tulajdonsága, hogy a titkosított szöveg hossza duplája a nyílt szöveg hosszának?
a) ElGamal
b) RSA
c) Massey-Omura
d) AES

T57. Az alábbiak közül melyik kriptorendszer biztonságát ássa alá, hogy teljesen védtelen a „man in the middle" támadással szemben?
a) ElGamal
b) RSA
c) Massey-Omura
d) AES

T58. Az alábbi feltételek közül melyik az, amelyik elvárás a nyilvános kulcsú kriptorendszerek esetében, de nem elvárás a privát kulcsú kriptorendszereknél?
a) Gyorsan elvégezhető titkosítása.
b) Gyorsan elvégezhető visszafejtés.
c) Magas biztonsági szint.
d) A titkosító kulcs ismeretében (további titok ismerete nélkül) ne leshessen emberi időn belül kiszámítani a visszafejtő kulcsot.

## 4.3 Kulcscsere RSA-val, diszkrét logaritmussal

A szimmetrikus titkosító algoritmusok nagyon gyorsak, lényegében késleltetés nélküli, valós idejű, kommunikációt tesznek lehetővé. Ezt a képességüket azonban csak azok a partnerek tudják kihasználni, akik rendelkeznek közös kulccsal. Meg kell oldani, hogy a kommunikálni szándékozó felek gyorsan hozzájuthassanak közös kulcshoz. ***A kulcscsere során tehát nyilvános csatornán keresztül szimmetrikus kulcsot osztunk szét a kommunikálni kívánó partnerek között.***

A legegyszerűbb megoldás egy megbízható kulcsosztó központ működtetése. Olyan ez, mint a laktanyaparancsnok volt katonakoromban, ő találta ki a jelszavakat és közölte parancsban az őrszolgálat tagjaival. Az következő algoritmusban feltételezzük, hogy Kriszta és Aladár külön-külön bizalmas kommunikációt tud folytatni a kulcsosztó központtal.

1. Kriszta és Aladár kér a kulcsosztó központtól egy alkalmi szimmetrikus kulcsot.
2. A kulcsosztó központtól generál egy alkalmi kulcsot, K-t.
3. A kulcsosztó központ a titkos csatornán egyszerre küldi el Krisztának és Aladárnak a közös K kulcsot.

A protokollban a kulcsosztó központ meghatározó szerepet játszik. A rendszer hibátlan működéséhez elengedhetetlen, hogy a kulcsosztó központ tisztességes legyen, a kérelmezőknek például ugyanazt a kulcsot küldje el; ne adja át harmadik félnek Kriszta és Aladár közös kulcsát, stb. Egy ilyen központ üzemeltetői a pszichológiai megtévesztés alapú támadás természetes alanyai. A rendszer nagyon érzékeny lenne a technikai hibákra és emberi mulasztásokra. További problémát jelent az, hogy Krisztának és Aladárnak is összeköttetésben kell állnia bizalmas csatornán keresztül a kulcsosztó központtal. Ilyen dedikált csatornák kiépítése lehetséges, de költséges. Aszimmetrikus titkosítás nélkül a kulcscserére csak költséges megoldások ismertek.

Aszimmetrikus titkosítás alkalmazásával a kulcscsere problémája könnyen megoldható a következő protokoll alkalmazásával. Abban $E$ és $D$ egy aszimmetrikus kriptorendszer kódoló és dekódoló függvénypárja (ld. 1.6. fejezet).

1. Kriszta és Aladár megbeszélik, hogy bizalmas üzeneteket akarnak cserélni.
2. Kriszta kikeresi Aladár nyilvános kulcsát – $N_A$ – egy nyilvános adatbázisból.
3. Kriszta generál egy alkalmi szimmetrikus kulcsot, $K$-t.
4. Kriszta az $N_A$ felhasználásával kiszámítja $T = E(K, N_A)$-t és elküldi $T$-t Aladárnak.
5. Amikor Aladár megkapja $T$-t, kiszámítja $D(T, T_A)$-t, ahol $T_A$a csak általa ismert titkos kulcs.

Ha minden szereplő korrekt, akkor a protokoll is az, azaz az 5. pontban kiszámított érték, $D(T, T_A)$, megegyezik $K$-val. A protokoll sajátossága, hogy tetszőleges aszimmetrikus titkosító algoritmussal működik. Látható továbbá, hogy a protokoll végrehajtásához semmilyen titkos csatornára nincs szükség. Van azonban egy apró hibája is, az hogy a $K$-t kizárólag Kriszta generálja, ami lehetőséget adhat neki csalásra.

Ezt a kis hiányosságot kiküszöböli a Diffie-Hellman kulcscsere protokoll. A protokoll végrehajtásához szükségünk van egy nagy prímszára, amelyet $p$-vel és egy primitív gyökre moduló $p$, amelyet $g$-vel jelölünk.

1. Kriszta és Aladár megállapodnak, hogy a kulcscseréhez a $p$ prímszámot és a $g$ primitív gyököt használják.
2. Kriszta választ egy $u$ véletlen számot $2$ és $p-2$ között.

3. Kriszta kiszámítja a $K_K = g^u \ mod \ p$ értéket és elküldi $K_K$ -t Aladárnak.
4. Aladár választ egy $v$ véletlen számot *2* és *p-2* között.
5. Aladár kiszámítja a $K_A = g^v \ mod \ p$ értéket és elküldi $K_A$ -t Krisztának.
6. Kriszta kiszámítja a $K = K_A^u \ mod \ p$ értéket.
7. Aladár kiszámítja a $K = K_K^v \ mod \ p$ értéket.

Látható, hogy ebben a protokollban sem használunk titkos csatornát. A protokoll korrektsége azon múlik, hogy hatványt úgy kell hatványozni, hogy a kitevőket összeszorozzuk. Ezen kívül csak azt kell tudni, hogy a hatványozás és a maradék kiszámítása tetszőleges sorrendben hajthatóak végre. Ezekből következik, hogy

$$K = K_A^u \ mod \ p = (g^v)^u \ mod \ p = g^{vu} \ mod \ p = (g^u)^v \ mod \ p = K_K^v \ mod \ p.$$

Ha tehát minden szereplő korrekt, akkor a protokoll végén Kriszta és Aladár tényleg rendelkezik közös kulccsal. Figyeljük meg, hogy $u$-t csak Kriszta, míg $v$-t csak Aladár ismeri, így a $K$ generálásához egyenlő arányban járulnak hozzá.

A protokoll biztonsága a diszkrét logaritmus kiszámításának nehézségén múlik. Egy hacker csak a $K_A$ és a $K_K$ értékeket ismerheti. Ezekből az értékekből, mai tudásunk szerint $K$-t csak az $u$-val, illetve a $v$-vel való hatványozással lehet kiszámítani. Őket azonban $K_K$-ból, illetve $K_A$-ból diszkrét logaritmus számítással lehet kinyerni.

### *Tesztkérdések:*

T59. A kulcscsere során _____ csatornán keresztül _____ kulcsot juttatunk el a kommunikálni kívánó partnereknek. Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: kedves, nyilvános, aszimmetrikus, megtévesztő, szimmetrikus.

T60. Kulcscserére azért van szükség

a) mert a partnerek elfelejtették megbeszélni a titkos kulcsot,
b) kicserélték a bejárati ajtó zárját,
c) az alkalmi titkos kulcsot a partnerek megismerjék.

T61. Az interneten kulcscserét

a) csak szimmetrikus titkosítással lehet megvalósítani.
b) csak aszimmetrikus titkosítással lehet megvalósítani.
c) csak Caesar titkosítással lehet megvalósítani.

T62. Mely állítások igazak az alábbiak közül: A Diffie - Hellman kulcscsere algoritmus

a) biztonsága a diszkrét logaritmus probléma nehézségén múlik,
b) a DES algoritmust használja.
c) nem biztonságos, nem szabad használni.
d) a két fél egyformán járul hozzá a titkos kulcshoz.
e) az egyik fél generálja a kulcsot és ElGamal titkosítással elküldi a másiknak.

## 4.4 Hibrid kriptorendszerek, SSL

Az internet és a mobiltelefon hálózatok, amelyek az elmúlt évtizedekben mindennapjaink részévé váltak, nyilvánosak. Viszonylag egyszerűen rá lehet kapcsolódni a fizikai hálózatra és így követni vagy tárolni lehet az azon átfolyó információt. Az internet hőskorában, a múlt század 90-es éveinek elején, széles körben elterjedt Netikett csak nyilvános információ továbbítását tekintette elfogadhatónak. Időközben az internet úgy elterjedt, hogy etikai alapon lehetetlen működtetni. Mindennapossá vált továbbá bizalmas adatok továbbításának az igénye is. Nem változott meg azonban az internet természete, ma is egy nyilvános hálózat.

Személyes és különleges adatokat, banki tranzakciókat, döntés-előkészítő anyagokat, választások részeredményét, stb. csak titkosítva lehet ilyen csatornán továbbítani. A korábbi fejezetekben példákat mutattunk be biztonságosnak tekintett titkosító módszerekre. Mindennapi használatra kielégítő megoldást azonban sem a szimmetrikus, sem az aszimmetrikus titkosítási módszerek sem nyújtanak. Az előbbiek elegendően gyorsak, de a partnereknek ismerni kell egy közös kulcsot, amelyet szimmetrikus titkosítási módszerekkel nem tudnak kicserélni. Az aszimmetrikus algoritmusoknak nincs szükségük kulcscserére, de több nagyságrenddel lassúbbak, mint a szimmetrikusok. Az előző fejezetben ismertetett kulcscsere algoritmusok teszik lehetővé, hogy a szimmetrikus és aszimmetrikus titkosító algoritmusok előnyös tulajdonságainak ötvözésével biztonságos és olcsó kommunikációs csatornákat lehessen kiépíteni nyilvános hálózatokon. Ezeket ***hibrid kriptorendszereknek*** nevezzük.

Eminens példa az SSL (Secure Socket Layer) protokoll, amelyet 1993-ban vezetett be a Netscape Communications. Azóta többször finomították és a biztonságos internetes kommunikáció szabványos eszközévé vált. Weblapok címe általában a ***http*** betűkombinációval kezdődik, amely a *Hypertext Transfer Protocol* kifejezés rövidítése. A http tulajdonképpen egy „inforációátviteli protokoll elosztott, kollaboratív, hipermédiás, információs rendszerekhez[12]". Ez nem alkalmas bizalmas információ átvitelre. A HTTP-t kiegészítve az SSL protokollal kapjuk a HTTPS *Hypertext Transfer Protocol Secure* protokollt, amelyik már kielégíti az ilyen igényeket is. A bizalmas információ átvitelre alkalmas weblapok címe a ***https*** betűkombinációval kezdődik (ld. 3. ábra).

A hibrid kriptorendszerek az internetes kommunikáció fontos alkotó részei. Nagyon leegyszerűsítve egy szimmetrikus titkosító algoritmusból és egy kulcscsere protokollból állnak. A protokoll fő lépései az alábbiak:

1. Kriszta és Aladár megbeszélik, hogy bizalmas üzeneteket akarnak cserélni.
2. Kriszta elindítja a kulcscsere protokollt, amelynek eredményeképpen mindketten rendelkeznek a közös szimmetrikus $K$ kulccsal.

[12] http://hu.wikipedia.org/wiki/HTTP

3. Kriszta és Aladár a *K* kulcsot használva bizalmas kommunikációt folytat.
4. Kriszta és Aladár megsemmisítik *K*-t.

A protokoll eleget tesz a fejezet elején megfogalmazott követelményeknek; ötvözi a szimmetrikus és aszimmetrikus kriptográfiai előnyeit. A kulcscsere során kevés adatot kell aszimmetrikus algoritmussal kódolni, így a 2. lépés gyorsan végrehajtható. Jelentősebb adatmennyiséget a 3. lépésben szimmetrikus titkosítási algoritmussal kódolunk, amely elég gyors. A módszer gyorsaságát statisztikai adatok helyett azzal lehet bizonyítani, hogy a felhasználói élmény nem csökken attól, ha a http helyett a biztonságos https attribútumú weblapot használnak.

Felhívjuk a figyelmet a 4. pont által előírt akcióra. Az alkalmi kulcsokat általában valamilyen álvéletlen módszerrel generálják (ld. 1.5. fejezet). Ha többször egymás után használnak egy ilyen kulcsot és a kódolt üzenetek egy kódfejtő kezébe kerülnek, akkor az a kódolt üzenetek statisztikai tulajdonságait kihasználva visszafejtheti az eredeti üzenet egyes darabjait, rossz esetben az egész üzenetet. Ezért a szimmetrikus kulcsok egyszeri használata ma előírás. Az alkalmi kulcs törlése, hasonlóan a többi alkotóelemhez, nem a felhasználó feladata. Egyedi alkalmazások esetén azonban figyelni kell az ilyen apróságokra is.

*Tesztkérdések:*

T63. A hibrid kriptorendszerek egy _____ titkosító algoritmusból és egy _____ protokollból állnak.Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: aszimmetrikus, megtévesztő, szimmetrikus, kulcscsere.

T64. Hibrid kriptorendszerre azért van szükség

a) mert a szimmetrikus és az aszimmetrikus titkosító algoritmusok külön-külön nem elégítik ki az internet igényeit.
b) a szimmetrikus titkosító algoritmusok nem elég biztonságosak.
c) az aszimmetrikus titkosító algoritmusok nem elég biztonságosak.

T65. Melyik protokoll használ hibrid kriptorendszert?

a) A http.
b) A https.
c) A digitális aláírás.

## 4.5 A digitális aláírás technológiája

A kriptográfia egyik legfontosabb és leggyakrabban alkalmazott terméke a digitális aláírás. Fontosságát mutatja, hogy szabályozására a Föld szinte minden országában törvényt hoztak. Hazánkban az új évezred második évében fogadta el az Országgyűlés a 2001. évi XXXV. törvényt

az elektronikus aláírásról. Jegyzetünkben nem a törvényt, hanem azokat a technológiákat ismertetjük, amelyek mai ismereteink szerint kielégítik a törvény előírásait.

Egy hagyományos dokumentumon ***az aláírás azt jelenti, hogy az aláíró ismeri a dokumentum tartalmát, azzal egyetért és az abban foglaltakat magára nézve kötelezőnek ismeri el.*** Aláírás mellett a hagyományos dokumentumokon dátum is szerepel, amelyik azt mutatja, hogy a dokumentum a dátum időpontjában már létezett, az aláírással együtt pedig azt, hogy a dátum időpontja óta nem változott meg. ***Az aláírásnak egyedinek, másolhatatlannak és a dokumentumtól elválaszthatatlannak kell lennie.***

A digitális aláírás szükségességét Diffie és Hellman fogalmazta meg a már idézett 1976-os dolgozatukban. Rámutattak arra is, hogy aszimmetrikus titkosító algoritmussal a digitális aláírás általában megvalósítható. Arra van csak szükség, hogy a titkosítás és a megfejtés algoritmusai felcserélhetőek legyenek. A 4.1. fejezetben tárgyalt RSA algoritmus egyszerű példát ad ilyen tulajdonságú kriptorendszerre.

A megfelelő kriptorendszer kiválasztása meghatározza az aláíró $D$ és az ellenőrző $E$ algoritmusokat. Ezek tulajdonképpen az aszimmetrikus titkosító rendszerek megfejtő, illetve tikosító algoritmusai. A kriptorendszer kiválasztása után a digitális aláírás három lépésből áll:

1) Paraméterek, azaz a titkos aláíró – $T$ - és a nyilvános ellenőrző – $N$ - kulcsok meghatározása.
2) Az $M$ dokumentum aláírása, azaz $M' = D(M,T)$ kiszámítása. Az $M$ és $M'$ nyilvánosságra hozatala.
3) Az aláírás ellenőrzése:
   3.1. $M'' = E(M',N)$ kiszámítása.
   3.2. Ha $M'' = M$, akkor elfogadjuk az aláírást, különben elutasítjuk azt.

Az aláíró, nevezzük Krisztának, az 1. lépésben generál egy csak általa ismert aláíró kulcsot és a hozzá tartozó ellenőrző kulcsot. Az előbbit úgy tárolja, hogy ahhoz csak egy általa teljes mértékben ellenőrzött berendezés férhessen hozzá. A kulcsgenerálás bonyolult matematikai feladat, a kulcsok biztonságos tárolása pedig komoly technikai probléma, amellyel az 5. fejezetben foglalkozunk.

Most inkább nézzük meg, hogy a séma, jó paraméterek választása esetén, tényleg biztosítja az aláírással szemben előbb megfogalmazott követelményeket. Az $M$ dokumentum aláírt példánya, $M'$, két szempontból is egyedi. Egyrészt azért, mert azt csak a Kriszta által ismert kulccsal hozták létre. Másrészt azért, mert ha az $M$-et Kriszta egy másik dokumentumra cseréli, akkor annak az aláírt példánya különbözik $M'$-től. Az $M$ és $M'$ tehát egymástól elválaszthatatlan párt alkot.

Az aláírás ellenőrzéséhez Kriszta titkos kulcsának nyilvános párját használjuk. Ha tehát a 3.1. lépésben valóban az $M'$-re alkalmazzuk az $E$ függvényt, akkor az $M$-et kell eredményül kapnunk és a 3.2. lépésben megállapíthatjuk, hogy a dokumentumot valóban Kriszta írta alá és az aláírás óta nem változott meg. Ha azonban a 3.1. lépésben kiszámított érték $M$-től különbözik, akkor vagy a dokumentum változott meg, vagy az ellenőrzött dokumentumot nem Kriszta írta alá.

A dátumozás elvégezhető hagyományos módon, a dokumentum elejére vagy végére írva. Az előre vagy hátra dátumozást ez persze nem védi ki. Az időbélyeg, amelyről az 5.4. fejezetben lesz szó, sokkal kényesebb igényeket is kielégít.

A fenti protokoll megfelel az aláírással szemben megfogalmazott jogi követelményeknek, technikai szempontból van azonban vele egy komoly probléma. Többször említettük már, hogy az aszimmetrikus algoritmusok lassúak. Márpedig néha terjedelmes dokumentumokat is alá kell írni. Ilyenekre az aláírás és az ellenőrzés is elfogadhatatlanul hosszú ideig tarthat.

Jogi és technikai szempontból is megfelelő aláíró protokollt egy kivonatkészítő eljárás, úgynevezett hash függvény beiktatásával kapunk. ***Egy hash függvény tetszőlegesen nagy dokumentumból fix hosszúságú kivonatot készít úgy, hogy gyakorlatilag lehetetlen két olyan dokumentumot konstruálni, amelyeknek a kivonata megegyezik.*** Ilyen függvényeket ***ütközésmentesnek*** nevezzük Gyorsan számolható és ütközésmentes függvényt nehéz konstruálni. Több, elterjedt hash függvényről derítették ki az utóbbi években, hogy nem elég biztonságosak. Ma már csak a SHA – Secure Hash Algorithm – függvénycsalád tagjait tekintik elegendően biztonságosnak.

Egy h-val jelölt hash függvény alkalmazásával a digitális aláírás protokoll fő lépései a következőek.

1. Paraméterek, azaz a titkos aláíró – *T* - és a nyilvános ellenőrző – *N* - kulcsok meghatározása.
2. Aláírás
   2.1. kivonatkészítés az *M* dokumentumból, azaz *KM = h(M)* kiszámítása.
   2.2. a *KM* lenyomat aláírása, azaz *KM' = D(KM,T)* kiszámítása. Az *M* és *KM'* nyilvánosságra hozatala.
3. Az aláírás ellenőrzése:
   3.1. *KM = h(M)* kiszámítása.
   3.2. *KM" = E(KM',N)* kiszámítása.
   3.3. Ha *KM" = KM*, akkor elfogadjuk az aláírást, különben elutasítjuk azt.

Ebben a protokollban nem az eredeti dokumentumra, hanem annak kivonatára alkalmazzuk az aláíró függvényt. A *h* ütközésmentessége miatt azonban gyakorlatilag lehetetlen *M*-et úgy módosítani, hogy annak kivonata megegyezzen *M* kivonatával. *M* és *KM'* tehát majdnem ugyanolyan elválaszthatatlan párt alkot, mint *M* és *M'*.

Az aláírás ellenőrzője is ismeri *h*-t, hiszen az egy szabványos függvény. A 3.1. lépésben tehát ő is ki tudja számítani *M* lenyomatát. ***Ha M az aláírás és az ellenőrzés között nem változott meg, továbbá Kriszta írta alá M-et és N a T nyilvános párja, akkor KM" megegyezik KM-mel és a 3.3. lépésben nyugodtan elfogadhatjuk az aláírást.*** Ha azonban valamelyik feltétel nem teljesül, akkor el kell utasítani azt, mert vagy a dokumentum változott meg, vagy nem Kriszta írta alá vagy nem az ő ellenőrző kulcsát használta az ellenőr.

A 2001. évi XXXV. törvény az elektronikus aláírásról megkülönböztet egyszerű, fokozott biztonságú és minősített elektronikus aláírást. A törvény technológia semleges. Egyszerű elektronikus aláírásként akár egy beszkennelt aláírás másolatai is megfelelnek. ***Mai ismereteink szerint fokozott biztonságú és minősített elektronikus aláírást csak aszimmetrikus kriptográfiai algoritmusokkal lehet realizálni.*** Másként nem biztosítható, hogy az aláírás „alkalmas az aláíró azonosítására, egyedülállóan az aláíróhoz köthető és a dokumentum tartalmához olyan módon kapcsolódik, hogy minden – az aláírás elhelyezését követően a dokumentumban tett – módosítás érzékelhető".

***Kriptográfiai szempontból nincs különbség a fokozott biztonságú és a minősített elektronikus aláírás között, ugyanazokat az algoritmusokat használják. A különbség az aláírást létrehozó eszköz elhelyezésében és minőségében van.*** Fokozott biztonságú az olyan aláírás is, amely saját számítógépen kerül létrehozásra. Ahhoz azonban, hogy minősített aláírást is létrehozhassak rajta, hitelesíttetni kell.

*Tesztkérdések:*

T66. Az aláírásnak _____, _____ és a dokumentumtól _____ kell lennie. Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: jól olvashatónak, egyedinek, digitálisnak, másolhatatlannak,aszimmetrikusnak, megtévesztőnek, elválaszthatatlannak.

T67. Mely állítások igazak az alábbiak közül: Kivonatkészítő, hash függvényre azért van szükség a digitális aláírás protokollban, mert

a) a szimmetrikus aláíró algoritmusok lassúak.
b) terjedelmes dokumentumokat is alá kell írni.
c) mert az aláíró algoritmusok aszimmetrikus titkosítást használnak, így lassúak.
d) a kivonat legyen független a dokumentumtól.
e) az aláírás független legyen a dokumentum tartalmától.

T68. Egy beszkennelt aláírás másolata

a) fokozott biztonságú elektronikus aláírás.
b) minősített elektronikus aláírás.
c) egyszerű elektronikus aláírás.
d) nem tekinthető elektronikus aláírásnak.

T69. Mely állítások igazak az alábbiak közül: Ha két dokumentum egyetlen szóban különbözik, akkor

a) a lenyomataik majdnem mindig megegyeznek.
b) a digitálisan aláírt példányaik megegyeznek.

c) a digitálisan aláírt példányaik mindig különböznek.
d) a lenyomataik különböznek, de a digitálisan aláírt példányaik megegyeznek.
e) a lenyomataik és a digitálisan aláírt példányaik is különböznek.

# 5 Nyilvános kulcs infrastruktúra

## 5.1 Kulcsok tárolása

***A nyilvános kulcs infrastruktúra a biztonságos internetforgalom működtetéséhez szükséges hitelesítő szervezetek világméretű hálózata.*** Angol nevének, public key infrastructure, rövidítése PKI. Gyakran csak így szoktak rá hivatkozni, a továbbiakban mi is így teszünk. ***Alapfeladata az aszimmetrikus titkosításhoz és a digitális aláíráshoz szükséges nyilvános kulcsok biztonságos tárolása, hitelesítése és gyors elérhetőségének a biztosítása.*** Járulékos feladatokat is ellát, mint kulcsgenerálás, archiválás, időpecsét készítése, stb. Egy jól működő infrastruktúrához hasonlóan diszkréten háttérbe vonulva működik, a felhasználók csak akkor érzékelik a létezését, amikor valamilyen probléma merül fel. Ahhoz hasonlóan, ahogy az ivóvíz sok kilométeres távolságból, bonyolult berendezéseken jut el a csapba, csak akkor tudatosul bennünk, ha nem folyik a csap, a PKI-t is csak akkor észleljük, ha nem működik a szolgáltatás.

Miért van szükség a PKI-re? Mint már rámutattunk a szimmetrikus kulcsok, ritka kivételtől eltekintve, egyszer használatosak. Rövid életük alatt a számítógép operatív memóriájában könnyen tárolhatóak, funkciójuk ellátása után pedig megszünnek. Ezzel szemben az aszimmetrikus kulcspárok életciklusa években mérhető. A kulcspár tárolása mérete és érzékenysége miatt komoly probléma. Íme egy konkrét példa RSA paraméterekre:

*n* = 414772321977333657874487172187859149073689002425796628314954650272854 97\\
70454222949655402671659687013200883189355778549945535771795017962727287639\\
34354219338490138275218429972680924234994226230924163523694697478547489279\\
37930172405561059537047898949584181708290701727417257018398596989407897294\\
312744764435041,
*A*= 90875624018074278551349028875985619139556287071913785891548801795442644\\
84787553188901220220999045702816955569519450722289385865572831061354826008\\
71781166392586295606683619638848439267360824078553718177003590710761496650\\
66365828559146726866299093554447846639957816374813968569815449423805053102\\
88916616779777,
*E* = 65537.

A példában *n* a modulust, *A* az aláíró kulcsot, *E* pedig az ellenőrző kulcsot jelenti. A sorok végén a \\ jel azt jelenti, hogy a szám a következő sorban folytatódik. A paramétereket a szerző generálta, *n* valóban két nagy prímszám szorzata. Mai ismereteink szerint kicsi a valószínűsége annak, hogy *n*-t

valaki tényezőkre tudja bontani. Az aláírónak A-t biztonságos helyen kell tárolnia[13]. Egy ilyen számszörnyeteget lehetetlen megjegyezni. Általában egy smart kártyára másolják, így a tulajdonosa mindig magánál tarthatja és szükség esetén egy számítógépbe helyezve könnyen használni tudja. Ezzel a titkos kulcs tárolását megoldottuk.

Mi történjen azonban a nyilvános kulccsal? Úgy kell persze tárolni, hogy bárki hozzáférjen. Van azonban még egy fontos kritérium: a potenciális partnernek vagy ellenőrzőnek biztosnak kell lennie abban, hogy a nyilvános kulcs valóban ahhoz a személyhez vagy szervezethez tartozik, amely adat a nyilvános adatbázisban szerepel. A probléma gyakorlati fontosságát egy példával szemléltetjük. Számítógépünkön gyakran kapunk olyan üzenetet, hogy valamelyik szoftverünknek, Windows, Adobe, stb. új verziója jelent meg. A frissítés ajánlott, célszerű végrehajtani. *Honnan tudjuk azonban azt, hogy valóban a nevezett szoftver hiteles verzióját tölti le a számítógépünk nem pedig egy meghekkelt változatot?* Utóbbi akár egy olyan néhány soros program is lehetne, amelyik letörli a számítógépen található összes adatot. A számítógépünk döntésében azért bízhatunk meg, mert „megtanították” a digitális aláírás ellenőrzésére. A frissítés mellé ugyanis a gyártó hiteles aláírást mellékel. A számítógép a PKI-t használva ellenőrzi a gyártó nyilvános kulcsának, majd az aláírásnak a hitelességét és csak akkor engedi meg a telepítést, amikor mindkettőt rendben találta.

### *Tesztkérdések:*

T70. A nyilvános kulcs infrastruktúra a _____ internetforgalom működtetéséhez szükséges _____ szervezetek világméretű hálózata.Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: egyedi, digitális, biztonságos, informatikai, megtévesztő, hitelesítő.

T71. Hol tárolhatóak a szimmetrikus titkosító kulcsok?

a) Smart kártyán.
b) Nem kell tárolni őket.
c) Számítógépen.
d) Memorizáljuk őket.

T72. Hol tárolhatóak az aszimmetrikus titkosító kulcsok?

a) Smart kártyán.
b) Nem kell tárolni őket.
c) Számítógépen.
d) Memorizáljuk őket.

[13] Bár n nyilvános, célszerű A-val együtt ugyanott tárolni, hiszen az aláíráshoz mindkét szám szükséges.

## 5.2 Hitelesítő szervezet

Nem egy, hanem a világon szétszórtan működő számtalan, hasonló funkciójú szervezetről van szó. A PKI csomópontjainak tekinthető bizalmi szervezetek. *Feladatuk a nyilvános kulcsok biztonságos menedzselése, beleértve a kulcsok hitelesítését, az automatikus hitelesítés ellenőrzés támogatását, időbélyeg szolgáltatás, stb.* A hitelesítő szervezetek fontosságát mutatja, hogy működésüket jogszabályok, hazánkban a 2001. évi XXXV. törvény az elektronikus aláírásról szabályozza. Az egyik legismertebb hitelesítő szervezet a Verisign, Inc, amelynek az USA-ban van a központja. Hazánkban a Netlock Kft. foglalkozik üzleti alapon hitelesítés szolgáltatással.

A nyilvános kulcsok menedzselése azok teljes életciklusára értendő. A nyilvános kulcsot, néhány kiegészítő adat társaságában, adatbázisokban tárolja. A kiegészítő adatok a kulcs tulajdonosára, felhasználásának körülményeire, érvényességi idejére, státuszára, stb. vonatkoznak. Az 5.4. fejezetben megadjuk a szabványos tanúsítvány adattartalmát. A hitelesítő szervezetnek legalább ezeket az adatokat tárolnia kell. Ebben a fejezetben csak néhány általános elvet fogalmazunk meg.

Kulcs tulajdonosa természetes és jogi személy is lehet, olyan adatokat kell róluk tárolni, amelyek alkalmasak az egyértelmű azonosításra. Az adatbázisban tárolni kell, hogy a kulcs milyen algoritmushoz készült és milyen célra (megfejtés, aláírás) használható. Az érvényességi idő a teljes életciklusnak csak egy része, azt jelenti, hogy annak letelte után már nem lehet a kulcsot elsődleges akcióra használni. A bonyolult megfogalmazás annyit jelent, hogy aláíró kulcsot nem lehet aláírásra, a megfejtő kulcsot pedig az érvényességi idő után titkosított dokumentum megfejtésére használni. Ugyanakkor korábban titkosított dokumentumot is meg kell tudni fejteni az életcikluson belül. Hasonló vonatkozik az aláírásra is. Lehetséges, sőt nagyon is életszerű, hogy egy kulccsal már nem lehet aláírni, de korábban készült aláírást ellenőrizni kell.

Az érvényességi idő az előfizetés idejétől, a kulcs vélelmezett biztonságától és előre nem látható tényezőktől függ. Amikor elhelyezünk egy kulcsot a hitelesítő szervezetnél, akkor vélelmezik, hogy az mennyi ideig tekinthető biztonságosnak. A vélelmezett idő letelte után vagy meghosszabbítják a kulcs használhatóságának időtartamát, vagy új, erősebb kulccsal cserélik ki. Ez és az előfizetés lejárta előre látható esemény. Fel kell azonban készülni olyan esetekre is, amikor a titkos kulcs megsemmisül vagy ellopják. Ilyenkor a kulcsot vissza kell vonni, tovább már nem használható. A felhasználónak új kulcspárt kell igényelni.

Bármilyen ok miatt is vonták vissza a kulcspárt, a hitelesítő szervezetnek a nyilvános kulcsokat, azok teljes életciklusában, tárolni kell. A nyilvános kulcsok akkor sem veszhetnek el, ha a hitelesítő szervezetet felszámolják. Ilyen esetben egy másiknak kell átvennie őket.

*Új felhasználók adatainak rögzítése és továbbítása a hitelesítő szervezet felé egy regisztráló egység feladata, amelyet néha regisztráló szervezetnek is neveznek*. A nyilvános kulcs hitelessége szintjétől függően az adatok rögzítése többféle módon történhet. A legegyszerűbb esetben a felhasználó interneten keresztül adja meg az adatait. Ezzel szemben fokozott biztonságú hitelesítés igénylése esetén az adatok helyességét közjegyző által hitelesített dokumentummal kell igazolni.

A regisztráció része a nyilvános kulcs felhasználóhoz való hozzárendelése, sőt a kulcspár generálása is. A szerző utóbbi gyakorlattal korábban nem értett egyet. Az aszimmetrikus titkosításra ugyanis csak akkor teljesül, hogy a titkos kulcsot csak a tulajdonosa ismerheti, ha ő maga generálta. Be kellett azonban látnia, hogy a technológia elterjedésével csak a felhasználók maroknyi kisebbsége képes biztonságos kulcspár generálására. A döntő többségnek elegendően biztonságosak a regisztrációs szervezetek által biztosított egyszerű kulcsgeneráló alkalmazások. Fontos azonban, hogy azok korrektségét és megbízhatóságát folyamatosan ellenőrizzék.

*Tesztkérdések:*

T73. A hitelesítő szervezet adatbázisában tárolni kell a _____ kulcs _____ adatait, a kulcs milyen _____ készült és milyen célra használható. Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: nyilvános, titkos, egyedi, tulajdonosának, készítőjének, algoritmushoz, aláíráshoz.

T74. Milyen feladatai vannak a hitelesítő szervezeteknek?

a) Az aláíró kulcsok tárolása.
b) A megfejtő kulcsok tárolása.
c) A megfejtő kulcsok hitelesítése.
d) Az aláíró kulcsok hitelesítése.

T75. Milyen feladatai vannak a regisztráló szervezeteknek?

a) Digitális aláíráshoz kulcspárok készítése.
b) Hitelesítést igénylők adatainak felvétele.
c) A megfejtő kulcsok továbbítása a hitelesítő szervezetnek.
d) Az aláíró kulcsok hitelesítése.

## 5.3 Tanúsítvány

***A tanúsítvány bizonyítja a kulcspár nyilvános tagja tulajdonosának kilétét. A tanúsítvány tulajdonképpen egy néhány kilobájtos, szabványos felépítésű fájl, amelyet a hitelesítő szervezet ad a felhasználónak.*** Tartalmazza a tulajdonos nevét és elérhetőségét, a kiállító szervezet nevét és internetes elérhetőségét, az érvényesség kezdetét és végét. A technikai adatok között tartalmazza a nyilvános kulcsot és azt, hogy a kulcs milyen algoritmushoz készült és hány bitből áll. Végezetül tartalmazza a kulcsnak a kiállító szervezet aláíró kulcsával készült aláírását. Utóbbi a kulcsból készített kivonatot és annak titkosított változatát jelenti. A tanúsítvány nyilvános adatokat tartalmaz, azt bárki megtekintheti.

***Feladat**: Nyisson meg egy biztonságos weboldalt és nézze meg a tanúsítványát.*

Amikor a felhasználó egy dokumentumot aláír, akkor az aláírásával együtt az ellenőrző kulcs tanúsítványát is elküldi az ellenőrnek. Az ellenőr a tanúsítványból meg tudja állapítani, hogy a dokumentum küldője és a tanúsítvány tulajdonosa azonos-e. Ellenőrizni tudja a nyilvános kulcs

hitelességét is. Ez a kulcs aláírásának ellenőrzésével történik. Az ellenőrnek ehhez be kell szereznie a tanúsítványt kiállító hitelesítő szervezet nyilvános kulcsát. Utóbbihoz is kell egy tanúsítvány, amelyet egy másik hitelesítő szervezet állít ki. Továbbfolytatva a gondolatot hitelesítő szervezetek végtelen láncolatához jutunk, ami persze a gyakorlatban nem működhet. A problémát úgy oldják meg, hogy vannak abszolút megbízhatónak tekintett, úgynevezett, gyökérhitelesítők. Országonként általában egy, állami tulajdonban lévő gyökérhitelesítő van, hazánkban ezt a szerepet a Közigazgatási Gyökér Hitelesítés-Szolgáltató (KGyHSz) látja el. A tanúsítványt aláírhatja a tulajdonosa is. Ilyen aláírást általában egyszerűnek tekintenek, a gyökérhitelesítő saját kulcsának önaláírása azonban minősített.

A tanúsítványok ellenőrzését a felhasználótól általában átvállalja a számítógépe, pontosabban annak operációs rendszere vagy böngészője. A gyors ellenőrzés érdekében a nagyobb hitelesítő szervezetek elhelyezik a nyilvános kulcsukat az operációs rendszerekben és böngészőkben. Ezeknek tehát általában nem kell a világhálót végigkutatni aláírások ellenőrzése miatt.

*Tesztkérdések:*

T76. A tanúsítvány egy néhány kilobájtos, szabványos felépítésű _____, amelyet a _____ szervezet ad a felhasználónak. Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: fájl, igazolvány, hitelesítő, regisztráló.

T77. Milyen adatokat tartalmaz a tanúsítvány?

a) Digitális aláíráshoz az aláíró kulcsot.
b) Digitális aláíráshoz a megfejtő kulcsot.
c) A kulcs tulajdonosának adatait.
d) A tanúsítvány kibocsátójának aláírását.
e) A kulcs tulajdonosának aláírását.

## 5.4 Időbélyeg

Dokumentumokon dátum is található. Ennek igen fontos szerepe lehet például akkor, amikor egy határidőre benyújtandó pályázatról van szó. A dátum, hozzágépelve a szöveghez a dokumentum, így az aláírás részévé válik és nem módosítható. Digitális dokumentumokhoz akár századmásodpercre pontos és hiteles időt lehet elválaszthatatlanul hozzárendelni. A számítógépnek, amelyen a dokumentum készül, nagyon pontos órája van. Ez azonban csak a relatív időt méri, a kezdő időpontot a felhasználó tetszése szerint állíthatja előre vagy hátra. A számítógép saját ideje tehát nem tekinthető hitelesnek. Olyan megoldást kell tehát találni a dátumozásra, amelyiket a felhasználó nem befolyásolhat. ***Az időbélyeg egy független és folyamatosan ellenőrzött szolgáltató által kiadott elektronikus dokumentum, amely tartalmazza a kibocsátás dátumát és idejét akár századmásodperc pontossággal és azt a szolgáltató saját aláírásával hitelesíti.***

Időbélyeg szolgáltatására általában a hitelesítő szervezetek is jogosultak. Ez természetes módon egészíti ki termékpalettájukat. Hogyan kerül azonban az időbélyeg a dokumentumra? Amikor a szolgáltatás előfizetője egy dokumentumra időbélyeget akar tetetni, akkor aláírja a dokumentumot és elküldi a hitelesítő szervezetnek. A szervezet ellenőrzi az aláírást, és ha rendben találja, akkor kiegészíti a dokumentum kivonatát a pontos idővel és aláírja azt a saját kulcsával. Végül az egészet visszaküldi az előfizetőnek. Időbélyeget természetesen csak már létező dokumentumra lehet tenni, így tökéletesen megfelel a dátum funkciójának.

Az időbélyeggel felszerelt dokumentum ellenőrzésének folyamatát alább ismertetjük. Az ellenőr rendelkezésére áll az eredeti dokumentum, M, annak a kivonata, KM, a kivonat szerző által aláírt változata, KM', a kivonat és a hozzá csatolt időbélyeg, KMI, valamint a KMI-nek az időbélyeg-szolgáltató által aláírt változata, KMI'. Az ellenőr először megvizsgálja, hogy a KMI-t valóban az időbélyeg-szolgáltató írta-e alá. Ha az aláírást rendben találja, akkor elfogadja, hogy az időbélyeg hiteles. Ezután ellenőrzi, hogy a KM valóban az M kivonata-e és, hogy a KM-et valóban a szerző írta-e alá. Ha minden ellenőrzés pozitív eredménnyel jár, akkor elfogadja, hogy a dokumentum hiteles és az, az időbélyegen szereplő időpontban már létezett. Amennyiben az ellenőr bármelyik lépésben hibát talál, elutasítja a dokumentum hitelességét. Ismételten hangsúlyozzuk, hogy az egész folyamat automatizált, azt sem az aláíró, sem az ellenőr nem befolyásolhatja.

*Tesztkérdések:*

T78. Az időbélyeg egy független és folyamatosan ellenőrzött szolgáltató által kiadott _____ dokumentum, amely tartalmazza a _____ dátumát és idejét akár századmásodperc pontossággal és azt a szolgáltató saját _____ hitelesíti. Egészítse ki az előbbi hiányos mondatot az alábbiak közül az oda illő szavakkal: kártya formájú, elektronikus, lejárat, kibocsájtás, aláírásával, igazolványával.

T79. Az elektronikus dokumentumra gépelt dátum nem hiteles, mert

a) nincs aláírva.
b) a számítógépen a dátumot át lehet állítani.
c) a számítógépek nem mérnek időt.

T80. Mikor hiteles az időbélyeg?

a) Ha ráragasztjuk a dokumentumra.
b) Ha a dokumentum készítője tette a dokumentumra és aláírta.
c) Ha időbélyeg kiadására felhatalmazott szervezet bocsájtotta ki és aláírta.

# 6 Tesztkérdések megoldásai

A jegyzetben található tesztkérdések megoldásait a jobb átláthatóság kedvéért táblázatos formában adjuk meg. Azon kérdések esetében ahol a válasz nem egy betű, vagy egy szám, hanem a táblázatos forma kereteit meghaladó méretű, a táblázatban egy * szerepel a megoldás helyett, de a táblázatok alatt megtalálható ezen feladatok megoldása is.

| T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T10 |
|---|---|---|---|---|---|---|---|---|---|
| 1101 | 151 | 4 | c) | d) | a), c) | 32 | 11 | 13 | 120 |

| T11 | T12 | T13 | T14 | T15 | T16 | T17 | T18 | T19 | T20 |
|---|---|---|---|---|---|---|---|---|---|
| * | c) | * | b),e) | * | H | G | a), d) | b) | c) |

| T21 | T22 | T23 | T24 | T25 | T26 | T27 | T28 | T29 | T30 |
|---|---|---|---|---|---|---|---|---|---|
| c) | a) | c) | b) | c) | b) | a) | a) | e) | c) |

| T31 | T32 | T33 | T34 | T35 | T36 | T37 | T38 | T39 | T40 |
|---|---|---|---|---|---|---|---|---|---|
| a) | d) | b) | b),d) | * | a) | * | * | b) | 3,1,2 |

| T41 | T42 | T43 | T44 | T45 | T46 | T47 | T48 | T49 | T50 |
|---|---|---|---|---|---|---|---|---|---|
| a) | a) | b),c) | * | b) | a) | a), c) | c) | b) | c) |

| T51 | T52 | T53 | T54 | T55 | T56 | T57 | T58 | T59 | T60 |
|---|---|---|---|---|---|---|---|---|---|
| a) | d) | b) | c) | b) | a) | c) | d) | * | c) |

| T61 | T62 | T63 | T64 | T65 | T66 | T67 | T68 | T69 | T70 |
|---|---|---|---|---|---|---|---|---|---|
| b) | a),d) | * | a) | b) | * | b),c) | c) | c),e) | * |

| T71 | T72 | T73 | T74 | T75 | T76 | T77 | T78 | T79 | T80 |
|---|---|---|---|---|---|---|---|---|---|
| b) | a),c) | * | a),d) | a),b),c) | * | a),c),d) | * | b) | c) |

T11: Nyissa meg a Számítógép mappát, majd kattintson a jobb egérgombbal a C: meghajtóra, végül a felkínált lehetőségek közül a Tulajdonságok opcióra. A felugró ablakban megtalálja a C:

meghajtó kapacitását. A sorozat további tagjait például a Windows számológépével, azt tudományos üzemmódban használva számíthatja ki.

T13: Az (U,K,T,E,D) ötöst kriptográfiai rendszernek nevezzük, ha U, K és T véges halmazok, az E és a D pedig leképezések.

T15: Aszimmetrikus kriptorendszerben a titkosító kulcs nyilvános, a megfejtő kulcs pedig titkos.

T35: pettyes, viselkedés alapú, tudás alapú, téglalap alakú, biometrikus, fényképes.

T37: A többcsatornás azonosítás azt jelenti, hogy az ellenőr több, független azonosítót kér.

T38: A hozzáférési jogosultság ideiglenes szüneteltetése munkaidőn kívül és szabadságon nem az azonosítás, hanem a jogosultságkezelés feladata.

T44: A pszichológiai manipuláció olyan megtévesztő módszerek megismerésével, leírásával és fejlesztésével foglalkozik, amelyek az emberek hiszékenységét, jóhiszeműségét és segítőkészségét használják ki.

T59: A kulcscsere során nyilvános csatornán keresztül szimmetrikus kulcsot juttatunk el a kommunikálni kívánó partnereknek.

T63: A hibrid kriptorendszerek egy szimmetrikus titkosító algoritmusból és egy kulcscsere protokollból állnak.

T66: Az aláírásnak egyedinek, másolhatatlannak és a dokumentumtól elválaszthatatlannak kell lennie.

T70: A nyilvános kulcs infrastruktúra a biztonságos internetforgalom működtetéséhez szükséges hitelesítő szervezetek világméretű hálózata.

T73: A hitelesítő szervezet adatbázisában tárolni kell a nyilvános kulcs tulajdonosának adatait, a kulcs milyen algoritmushoz készült és milyen célra használható.

T76: A tanúsítvány egy néhány kilobájtos, szabványos felépítésű fájl, amelyet a hitelesítő szervezet ad a felhasználónak.

T78: Az időbélyeg egy független és folyamatosan ellenőrzött szolgáltató által kiadott elektronikus dokumentum, amely tartalmazza a kibocsátás dátumát és idejét akár századmásodperc pontossággal és azt a szolgáltató saját aláírásával hitelesíti.

# 7 Kiegészítő irodalom

[1] Johannes Buchmann: Introduction to cryptography, Second edition. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 2004.

[2] M. Bellare and P. Rogaway:The exact security of digital signatures - How to sign with RSA and Rabin. Berlin : Springer-Verlag, 1996., Eurocrypt '96, LNCS, 1070. 399-416.

[3] Budai Balázs Benjámin: E-Government, avagy kormányzati és önkormányzati kihívások az on-line demokrácia korában, Aula Kiadó, 2002.

[4] Buttyán Levente és Vajda István: Kriptográfia és alkalmazásai, Typotex, 2004.

[5] Catalano, Dario:Contemporary Cryptology. : Springer, 2005.

[6] Joan Daemen and Vincent Rijmen:The Design of Rijndael: AES - The Advanced Encryption Standard, Springer Verlag, 2002.

[7] Folláth János, Huszti Andrea és Pethő Attila: Informatikai biztonság és kriptográfia, e-jegyzet, Debreceni Egyetem, 2010.

[8] John M.D. Hunter: An information security handbook, Springer, 2001.

[9] Ködmön József:Kriptográfia: Az informatikai biztonság alapjai, a PGP kriptorendszer használata, ComputerBooks, 1999/2000.

[10] A.J. Menezes, P.C. van Oorschot and S.A. Vanstone: Handbook of applied cryptography, CRC, 1996.

[11] Kevin D. Mitnick és William L. Simon: A legendás hacker: A behatolás művészete, Perfact Kiadó, 2006.

[12] Josef Pieprzyk, Thomas Hardjono and Jennifer Seberry: Fundamentals of Computer Security, Springer, 2003

[13] B. Schneier: Applied cryptography: protocols, algorithms and source code in C, 1996.

[14] Stinson, Douglas R.:Cryptography. Theory and practice. Third edition. Discrete Mathematics and its Applications (Boca Raton). *Chapman & Hall/CRC, Boca Raton, FL,*2006.

Nemzeti
Közszolgálati Egyetem
Vezető- és Továbbképzési Intézet

BESZÉDES ÁRPÁD – GERGELY TAMÁS

# Alkalmazásbiztonság

Budapest, 2014

A tananyag az ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel című projekt keretében készült el.




**Szerző:**
© Beszédes Árpád, Gergely Tamás 2014

**Kiadja:**
© NKE, 2014

**Felelős kiadó:**
Patyi András
rektor

# TARTALOM

# 1. BEVEZETŐ

Az alkalmazásbiztonság egy egyszerű és egyben mégis összetett fogalom. Egyszerű, hiszen kifejezi az alkalmazás azon tulajdonságát, hogy az védett legyen a különféle rosszindulatú támadásokkal szemben. Ugyanakkor összetett, hiszen nagyon sokféle támadás létezik, amelyek ellen más-más módon lehet és kell védekezni. Nyilvánvalóan más módszerrel védekezünk az átküldött adat illetéktelenek által való megismerése, mint mondjuk egy szolgáltatás ellehetetlenítését célzó „deny of service" támadás ellen.

Látni kell, hogy az alkalmazásbiztonság igen nagy hatással van a szoftver egészére. Egy szoftvert nem lehet utólag egy-két plusz modullal biztonságossá tenni. Ha nem gondolunk a biztonságra már a fejlesztés kezdetén, akkor ugyan utólag befoltozhatunk egy-két lyukat, de a szoftver soha sem fogja elérni a megfelelő biztonsági szintet. Nagyon fontos tehát, hogy a biztonságra már a fejlesztési projekt elején, a tervezésnél odafigyeljünk. A biztonsági szempontoknak és igényeknek tehát meg kell jelenniük a szoftverkövetelmények között. A másik sarkalatos pont a fejlesztés során a tesztelés. Nem csak arról van szó, hogy az egyes biztonsági követelmények megvalósítását hogyan és milyen módon lehet letesztelni, de a teszteknek nem csak a leírt követelmények megvalósítására, hanem a követelmények teljességére, pontosságára, szükségszerűségére is ki kell terjedniük. És persze a projekt érdekében az sem árt, ha mindezt hatékonyan tudjuk elvégezni, és a tesztelés eredményét a fejlesztés valóban fel is használja, azaz megfelelő módon kijavítja a szoftvert.

Ez a tananyag az alkalmazásbiztonság fentebb említett aspektusait ismerteti és részletezi. A 2. fejezetben bemutatjuk, hogy a szoftverbiztonság hogyan vonul végig a teljes szoftverfejlesztési életcikluson, hogyan illeszkedik az egyes fejlesztési modellekhez, illetve foglalkozunk a kockázat fogalmával is. A 3. fejezetben a követelmények meghatározásának és dokumentálásának módszereivel foglalkozunk, különös tekintettel a biztonsági követelményekre. A 4. fejezet a megbízható és hatékony teszt menedzsmentről szól, az 5., 6. és 7. fejezetek pedig különféle tesztelési módszereket mutatnak be, melyek segítségével a szoftver biztonsága hatékonyan verifikálható és validálható. A 8. fejezet olyan további témaköröket dolgoz fel, mint a szoftverek biztonságos üzemeltetése, megfigyelése, a szoftver fejlődése, a biztonsági rések befoltozását elősegítő megbízható és hatékony incidens menedzsment illetve a különféle szerződések alkalmazásbiztonsági szempontjai.

# 2. ALKALMAZÁSBIZTONSÁG A SZOFTVERFEJLESZTÉSI ÉLETCIKLUSBAN

## 2.1 Biztonsági szempontok a teljes szoftverfejlesztési folyamatban

A következő évtizedben nagyon valószínűtlen, hogy bárki aki részt vesz a szoftverfejlesztésben figyelmet ne kellene, hogy fordítson a biztonsági szempontokra. Ha egy biztonságos, és a támadásoknak ellenálló szoftvert akarunk készíteni, akkor a kezdetektől foglalkozni kell a biztonság kérdésével. Ez magában foglalja a titkosítás, sértetlenség, elérhetőség, azonosítás, engedélykezelés, rendelkezésre állás illetve a session-ök, kivételek, hibák és a konfigurációs paraméterek kezelésének kérdéseit. A folyamatok és a technológia elemeinek fejlesztése során a teljes szoftverfejlesztési ciklusban szem előtt kell tartani ezeket a szempontokat.

### 2.1.1 Szoftver biztonsági követelmények

Erős alapok nélkül egy ház is összeomlik, így van ez a szoftverek területén is.

A szoftverbiztonsági követelmények hiánya manapság a legtöbb fejlesztett szoftvernél megfigyelhető. Rendkívül fontos pontosan megtervezni, megfogalmazni és megvalósítani a biztonsági követelményeket, mert enélkül a szoftver nemcsak rossz minőségű lesz (mint egy „általános" követelmény hiánya esetén), de kifejezetten sérülékeny is, ami nagyságrendekkel komolyabb károkat okozhat a használóinak.

### 2.1.2 Biztonságos Software Design

A biztonsági kérdésekkel már a fejlesztés korai szakaszában érdemes és kell is foglalkozni.

Alapvető elv, hogy ha a szoftveren változtatnunk kell, akkor ez annál költségesebb, minél előrébb tartunk a megvalósításban. Egy követelmény-módosítást ugyanis annál több munkaterméken kell keresztülvezetni. Olyan ez, mint amikor egy épület tervezésénél nem gondolnak előre a tetőre telepítendő légkondicionáló berendezések súlyára, utólag sokkal költségesebb megerősíteni a szerkezetet, mint amennyivel a gondos tervezés és kivitelezés többe került volna.

Érdemes tehát a követelmények meghatározásakor biztonsági tervezési elveket követni (mint például a legkevesebb jogosultság vagy a feladatok szétválasztása). A tervezőknek és fejlesztőknek ismerniük kell ezeket a tervezési mintákat és a szoftvernek illetve projectnek megfelelőt kell választani ahhoz, hogy a szoftver a fontosabb biztonsági kockázatok ellen fel legyen készítve.

### 2.1.3 Biztonságos Software Implementation/Coding

„Biztonságos kódot írni" ez az egyik legfontosabb része a biztonságos szoftverfejlesztésnek.

Tisztában kell lennünk a biztonsági megoldások előnyeivel és hátrányaival. Az a kód, ami mellőz mindenféle biztonsági megoldást, nagyon könnyen támadható.

Néhány az elterjedt támadási módok közül:

- injection
- cross site scripting (XSS)
- cross-site request forgery (CSRF)
- buffer overflow.

### 2.1.4 Secure Software Testing

Nem lehet elég sokat tesztelni ahhoz hogy egy szoftvert teljesen biztonságosnak lehessen mondani.

A folyamatosan változó támadások ellen a program frissítésével lehet védekezni. Meg kell érteni, hogyan és mit kell tesztelni, milyen fenyegetések vannak, és ismerni kell a különféle tesztelési technikákat (mint például fekete do-

boz, fehér doboz tesztelés, logikai vizsgálat, penetrációs teszt, szimulációs tesztelés, regressziós tesztelés) ahhoz, hogy ezek kombinációjával hatékonyan tesztelhessük a szoftvert.

### 2.1.4 Szoftver elfogadás

Mielőtt a szoftver megjelenne vagy a fejlesztés következő szakaszába érne, meg kell felelnie az előre definiált különböző követelményeknek (ezek lehetnek minőségi, funkcionális, garanciális követelmények). Ezt egyfajta végső, a teljes rendszert üzemelés közben vizsgáló átvételi teszttel biztosíthatjuk. Törekedni kell arra, hogy az éles környezethez minél hasonlóbb rendszeren teszteljük, és ezen a rendszeren is meg kell, hogy feleljen a vele szemben megfogalmazott kritériumoknak.

### 2.1.5 Szoftvertelepítés, üzemeltetés, karbantartás

Amennyiben az ügyfél egy elfogadási teszt után elégedett a szoftverrel, a szoftvert telepíteni kell az ügyfélnél a biztonságot szem előtt tartva, különben az összes eddigi erőfeszítés amit a biztonság érdekében tettünk kárba veszhet. Miután sikeresen telepítettük a szoftvert, folyamatosan ellenőrizni kell, hogy a továbbiakban is megfelelően működjön, illetve javítani, ha bármi féle incidens bekövetkezne.

### 2.1.6 Szállítási lánc és szoftver beszerzése

A növekvő termelékenység miatt fontos megértenünk, hogy nem mindig tudunk mindent magunk megcsinálni és szükségünk lesz beszállítókra. Ezeknek megbízható forrásoknak kell lenniük, fontos szempont itt is a biztonság. Fontos, hogy a megfelelően értékeljük a beszállítókat, a szerződést technikailag ellenőrizzük. Ha kiválasztottuk a szállítót, gondoskodnunk kell a kód biztonságáról, például ellenőrző kóddal vagy érvényesség hitelesítésével.

***Fogalmak a 2.1. fejezetben:*** titkosítás, session, kivétel, SQL injection, cross site scripting (XSS), cross-site request forgery (CSRF), buffer overflow, fekete doboz tesztelés, fehér doboz tesztelés, penetrációs teszt, szimulációs tesztelés, regressziós tesztelés, elfogadási teszt.

## 2.2 V-modell

A szoftverfejlesztés egy strukturált és módszeres folyamat. A szoftverfejlesztési életciklusban ezt kisebb részekre, részfolyamatokra bontják és ezeket valósítják meg szekvenciálisan vagy párhuzamosan.

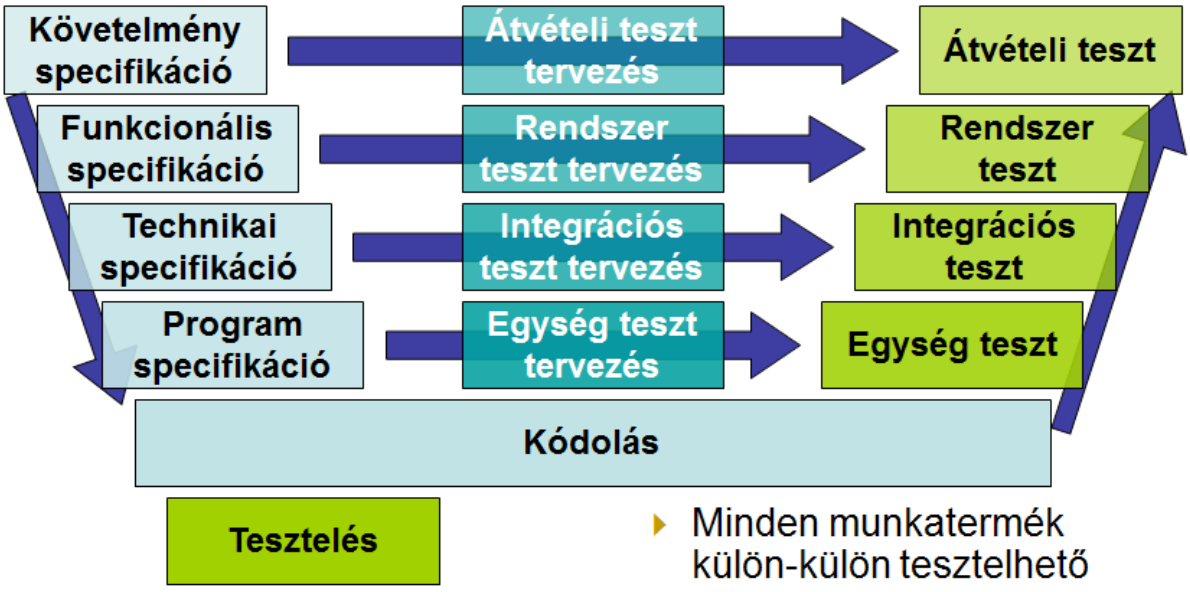
Kétféle főbb fejlesztési modellt különböztethetünk meg: a **lineáris** és az **iteratív** modelleket.

A V-modell egy lineáris modell, ami a korábbi vízesés modellre épül. A vízesés modell egyike a legtradicionálisabb szoftverfejlesztési modelleknek. Ez egy magasan strukturált, lineáris modell minek pontosan előredefiniált szakaszai vannak. Ebben a modellben ahhoz, hogy továbblépjünk a következő szakaszra be kell fejeznünk az aktuálisat. Ahogy a víz is csak egy irányba tud folyni, ha egy szakaszt lezártunk, akkor már nem léphetünk vissza. Éppen ezért nagyon fontos, hogy minden lépésben szem előtt tartsuk a biztonsági szempontokat is. Fontos, hogy már a követelmények elemzése és meghatározása közben gondoljunk a biztonsági szempontokra, és azt végigvigyük minden fázison, különben a későbbiekben komoly többletköltségink jelentkezhetnek.

A V-modell szintén a korai modellek családjába tartozik, amelyet a német védelmi minisztérium fejlesztett ki, majd főleg a német hadsereg szoftverfejlesztéseiben vált használatossá a továbbiakban. Maga az elnevezés nemcsak életciklus modellt, hanem egy teljes módszertant jelöl. A V-modell életciklus elképzelése nemcsak az egyes fázisok időbeli sorrendjéről szól, hanem arról is, hogy az egyes fázisokban mely korábbi fázisok termékeit kell felhasználni; illetve az adott fázis tevékenységét és termékét mely korábbi fázisban leírt követelmények, illetve elkészített tervek alapján kell ellenőrizni.

A V-modell használata főleg a biztonságkritikus számítógéprendszerek fejlesztése esetében terjedt el, ahol a szigorú szabályozási környezet miatt a teljes specifikáció már a tervezési fázisban ismert, és előreláthatólag nagyon kevés változás lesz menet közben. Ilyenkor egyértelmű, hogy minden követelmény, így a biztonsági kérdések is már a tervezési fázisban megjelennek, és azokat lépésről lépésre végig lehet vinni a teljes fejlesztési folyamaton.

A V modell lépései a következők:
- **Követelmény specifikáció**: a fejlesztési folyamat kiindulási pontját képező követelmények feltárása.
- **Funkcionális specifikáció**: a funkcionális követelmények (beleértve a biztonsági követelmények funkcionális aspektusait is) együttese alkotja. Mindezen specifikáció alapján megkezdhető a teljes rendszer konkrét tervezési folyamata.
- **Technikai specifikáció**: Technikai specifikáció fázisban a fejlesztési folyamatot további kisebb részekre, úgynevezett modulokra bontjuk fel a tervezési folyamat egyszerűsítése, áttekinthetőbbé tétele végett. A tervezés eredményeként a szoftver modulok specifikációja, valamint a köztük levő kapcsolódási folyamatok terve készül el. Tulajdonképpen ez a fázis felelős a szoftver architektúrájának kialakításáért.
- **Program-specifikáció**: Az egyes programegységek, modulok részletes, funkcionális leírása.
- **Kódolás**




*1. ábra V-modell és a hozzá tartozó tesztelési fázisok*

Továbbra is igaz, hogy az előző munkafázist be kell fejezni a következő lépés előtt.
A specifikációk átnézésével ellenőrizhető:
- Megfelel-e az előző munkaterméknek?
- Elég részletes-e a következő munkatermék helyes megvalósításához?
- Elegendő információt tartalmaz-e a munkatermék teszteléséhez?

Így minden munkatermék külön-külön tesztelhető.
Minden fázissal párhuzamosan kezdődik a tesztek tervezése is.

A biztonsági követelményeket az egyes szinteken is tesztelhetjük:
- **Egység teszt, program-specifikáció**: a statikus analízis, a kód felülvizsgálata ebben a részben elvégezhető, itt a forráskódban levő biztonsági rések kereshetők.
- **Integrációs teszt, technikai specifikáció**: mivel ebben a fázisban az egyes modulok közötti kapcsolat, valamint azok együttese kerül tesztelésre, az egyes modulok közötti kommunikáció biztonsága, az egyes osztályok, modulok szükséges láthatósága vizsgálható biztonság szempontjából.
- **Rendszer teszt, funkcionális specifikáció**: a rendszerteszt a feketedoboz tesztelésének kategóriájába esik. Ezzel az 5. fejezetben foglalkozunk részletesen.
- Átvételi teszt, **követelmény specifikáció**: ebben a fázisban éles környezetben kell vizsgálunk az elkészült termék biztonsági funkcióit.

2. Rakja helyes sorrendbe a V-modell lépéseit! (***a, b, c, d, e***)
   *a)* Követelmény specifikáció
   *b)* Funkcionális specifikáció

*c)* Technikai specifikáció
*d)* Program specifikáció
*e)* Kódolás

3. Mi nem ellenőrizhető a specifikációk felülvizsgálatával?
*a)* Megfelel-e az előző munkaterméknek?
*b)* Elég részletes-e a következő munkatermék helyes megvalósításához?
*c)* Elegendő információt tartalmaz-e a munkatermék teszteléséhez?
*d)* Helyes-e a megvalósítás?

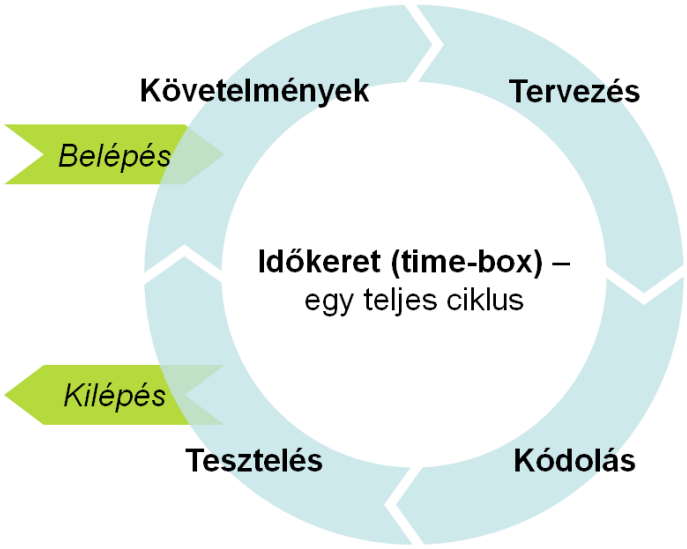
4. Párosítsa össze az egyes fejlesztési és tesztelési szinteket!

| | |
|---|---|
| 1. Funkcionális specifikáció | Integrációs teszt |
| 2. Követelmény specifikáció | Egység teszt |
| 3. Program-specifikáció | Átvételi teszt |
| 4. Technikai specifikáció | Rendszer teszt |

***Fogalmak a 2.2. fejezetben:*** lineáris fejlesztési modell, iteratív fejlesztési modell, követelmény specifikáció, funkcionális specifikáció, technikai specifikáció, program specifikáció.

## 2.3 Iteratív modell

Az iteratív modellekben a feladatok sokkal kisebb részekre vannak bontva, mint a V-modellben. Az elsődleges előnye ennek a modellnek, hogy sokkal több a kommunikáció a megrendelő és a fejlesztők között, ezzel sok időt és pénzt lehet spórolni ha pontosan tudjuk, hogy mit szeretne az ügyfél. A követelményeket nem szükséges rögtön a fejlesztési folyamat legelején véglegesíteni, így a tervezés, kódolás hamarabb elkezdődhet. Ehelyett, az egyes iterációk egyre jobban finomítják a követelményeket, amíg el nem jutunk a végső termékhez. A modellek egy-egy iterációja tulajdonképpen egy mini V-modellnek felel meg.




*2. ábra Iteratív modellek általános lépései*

Iteratív modelleknél előfordulhat, hogy a biztonsági kérdéseket (a rövid iterációk miatt) elhanyagolják az elején ezért fontos,hogy akkor is gondoljuk végig a biztonsági kérdéseket, ha azokat nem rögtön az első iterációk valamelyikében fogjuk megvalósítani. A biztonsági követelmények átfogóak, így különösen fontos őket már az első iterációkban is fejben tartani, még ha a megvalósításuk később lesz is.

### 2.3.1 Spirál modell

Egy fajta iteratív modell a **spirál modell**. A jellegzetessége ennek a modellnek, hogy minden fázisban van egy kockázatértékelési review tevékenység. Legfontosabb jellemzők:

- A spirál modell iterációkból áll, melyek folyamatosan ismétlődnek a projekt során.
- Valamennyi iteráció ugyanazon lépésekből áll.
- Lehetővé teszi a kockázatok korai felismerését.
- A megrendelőt minden fázisba aktívan bevonja.

### 2.3.2 Agilis fejlesztési folyamat

Az **agilis fejlesztési folyamat** a legelterjedtebb napjainkban. Az agilis fejlesztési modell az iteratív modellre alapoz, a célja, hogy minimalizálja a sikertelenség valószínűségét rövid fejlesztési szakaszokkal. Minden ilyen szakasz egy teljes szoftverfejlesztési modellt valósít meg. Egy nagy előnye ennek a modellnek, hogy a változások nem okoznak fennakadást, és könnyen, gyorsan megvalósíthatók.

### 2.3.3 RAD (Rapid Application Development)

A RAD fejlesztés során a prototípusok fejlesztését a lehető leggyorsabban valósítják meg. A felhasználókat bevonják a tesztelésbe. Rövid időkereteket szabnak minden feladatnak, amiket be kell tartaniuk. Továbbá fontos, hogy az emberek egy térben dolgozzanak.

### 2.3.4 Scrum (egy agilis szoftverfejlesztési modell)

A Scrum modellben pár hetes fejlesztési ciklusok (sprintek) vannak. Naponta maximum negyedórás megbeszéléseket tartanak. Gyakran önszerveződőek a fejlesztőcsapatok. Gyakori a páros programozás is.

***Melyik modellt válasszuk?***
Valójában a legcélravezetőbb a gyakorlatban, ha nem ragaszkodunk egyik modellhez sem szorosan, hanem két vagy több modell egy bizonyos kombinációját használjuk mindig a projektnek megfelelően. Fontos felismerni, hogy egyik szoftverfejlesztési modell, vagy modellek kombinációja sem hozhat létre eredendően biztonságos szoftvert. A szoftvert biztonságosnak kell tervezni, a fejlesztési folyamat minden részébe be kell építeni a biztonsági kérdésekkel kapcsolatos fejlesztéseket, teszteket.

***Fogalmak a 2.3. fejezetben:*** iteratív modell, spirál modell, review, agilis fejlesztés, páros programozás.

## 2.4 Kockázatkezelés szerepe

Ha nem lennének biztonsági kockázatok, akkor nem kellene a szoftverbiztonsággal foglalkozni. De vannak ilyenek. Teljesen biztonságos szoftvert lehetetlen készíteni, ehelyett a szoftverrel és használatával kapcsolatban felmerülő biztonsági kockázatokat kell különféle módokon kezelni.

De mi is az a kockázat? Ehhez ismerkedjünk meg pár fogalommal.

A **sebezhetőség** (vulnerability) egy olyan gyenge pont, amit a támadók kihasználhatnak. Például egy folyamat sérülékenysége, ha a belépés vagy kilépés nem megfelelően szigorú, illetve hiányos, gyenge beléptetési mechanizmus a fontos adatok hozzáféréséhez. Egy sebezhetőségi pont az is, ha a szoftver elfogad bármilyen felhasználótól bármilyen adatot előzetes ellenőrzés nélkül, vagy túl sok információt szolgáltat egy meghibásodás esetén a log fájlban.

A szoftverek sebezhetőségük miatt **fenyegetéseknek** (Threat) vannak kitéve. Ez csupán a lehetősége annak, hogy nem várt káros esemény bekövetkezik. Amikor ténylegesen be is következik a fenyegetés, az eredmény egy incidens. A támadók a **támadásoknál** (Attack) ezeket a gyenge pontokat használják ki. Általában a támadók ismerik a gyenge pontokat és ezeken keresztül próbálnak bizalmas adatokhoz hozzáférni vagy károsítani a rendszert.

A **valószínűség** (Probability) „likelihood”-ként is ismert. Annak az esélye hogy egy esemény bekövetkezik. A kockázatkezelés célja hogy lecsökkentsük egy káros esemény bekövetkezésének és az ebből adódó kárnak a valószínűségét egy előre meghatározott elfogadási szint alá.

A **hatás** (Impact) a fenyegetés tényleges bekövetkezése után lép fel. Változatos lehet, egy egyszerű meghibásodott alkatrésztől, a teljes cég összeomlásáig terjedhet.

Az **Exposure Factor** meghatározza a fenyegetés által okozott károkat. Fontos szerepet játszik a kockázatok számításánál. Ugyan fennáll a lehetősége egy támadásnak, de ha a tervezésnél, fejlesztésnél szem előtt tartották a biztonsági kérdéseket, akkor az esély a sikeres támadásra alacsony, ezáltal a támadás Exposure Factora alacsony lehet.

A **teljes kockázat** (Total risk) egy becslés egy káros hatás előfordulására. Ez a teljes kockázata a rendszernek minden biztonsági intézkedés nélkül. Általában 3 részből álló skálát használunk (alacsony, közepes, magas).

A biztonsági intézkedések után maradó kockázatot **megmaradó kockázatnak** (Residual Risk) hívjuk.

Biztonsági kockázat nemcsak az IT eszközök meghibásodási kockázata lehet, hanem az egyéb forrásokból eredő kockázat is például a megbetegedések. Fontos, hogy a kockázatkezelés segítse, és ne hátráltassa a fejlesztés menetét.

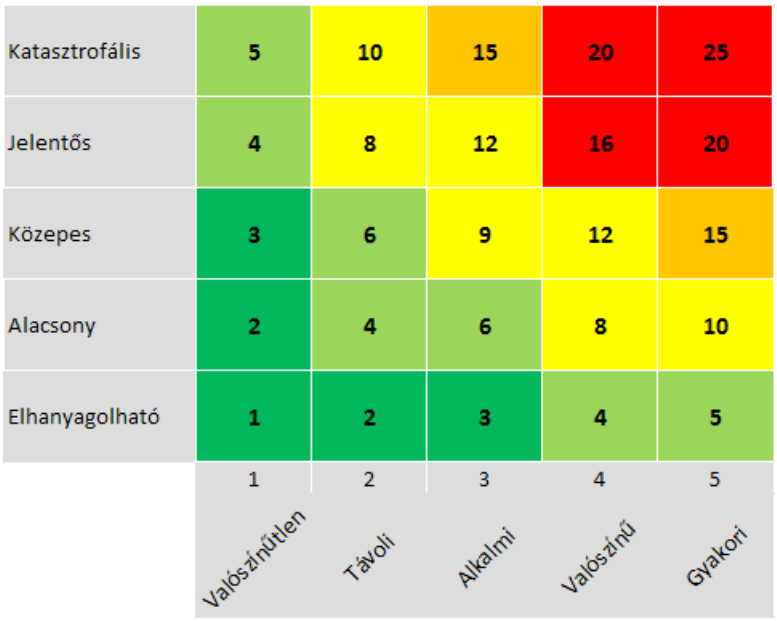
A kockázatkezelési folyamatok közé tartozik, hogy előzetesen értékeljük, hogy mire van szükség a teljes munkafolyamat során (biztonsági ellenőrzések, azonosítás, fejlesztés, tesztelés), hogy az esetlegesen adódó bekövetkezett meghibásodás, fenyegetés ne okozzon nagy fennakadást, ezért priorizálni kell a kockázatokat a bekövetkezés valószínűsége és az okozott kár figyelembevételével.

***Fogalmak a 2.4. fejezetben:*** kockázat, kockázatkezelés, sebezhetőség, fenyegetés, incidens, támadás, valószínűség, Exposure Factor, teljes kockázat, megmaradó kockázat.

## 2.5 Kockázatok felmérése

A szoftverrel és használatával kapcsolatban felmerülő biztonsági kockázatok kezeléséhez elengedhetetlen a lehetséges kockázatok felmérése.

A kockázatkezelésnek komoly kihívásokkal kell szembenéznie, mert még nem teljesen kiforrott tudomány, ezért sok olyan eset fordul elő, ami még ismeretlen, és ezek megoldása nehézkes lehet. Továbbá a szoftverek és az eszközök értéke szubjektív, a vállalat állapítja meg. Az adatok értéke például, amivel a szoftver dolgozik csak becslése a lehetséges veszteségnek. Tehát egy adatbázis támadása esetén nem tudjuk megmondani pontosan, hogy mennyi kárt fog okozni a támadó. Így nem állnak rendelkezésre pontos adatok egy adott hatás, annak bekövetkezése és az okozott kár tekintetében. A technikai biztonsági kockázat csak egy része a szoftver biztonsági követelményeinek.




*3. ábra Kockázatok lehetséges előfordulási gyakorisága és következményeinek súlyossága*

A kockázatot hagyományosan a lehetséges támadás valószínűsége és az okozott kár alapján számoljuk. Ezek viszont általában szubjektív becslések, mert ritkán állnak rendelkezésre pontos adatok. Ezért szokás inkább valószínűség és hatás szerint kategóriákba sorolni az egyes kockázatokat, és ez alapján meghatározni azok kockázati szintjét (például az 1. ábra szerinti módon).

A kockázat kiszámítása meglehetősen összetett lehet egy bonyolultabb szoftver esetén. Fontos ismernünk a következő fogalmakat: **SLE** (Single Loss Expectancy) az egyszeri várható veszteség, **ARO** egy adott fenyegetés miatt várhatóan egy év alatt bekövetkezett károk száma, **ALE** (Annual Loss Expectancy) éves várható veszteség. SLE-t a hatás és a bekövetkezés valószínűségének a szorzataként adjuk meg.

*SLE = Hatás x Valószínűség*

ALE egy egész évre vonatkozó mutatója a kockázatnak. Az egyszeri veszteség (SLE) és az adott veszteséget okozó esemény egész évben várható bekövetkező száma szorzataként adjuk meg:

*ALE = SLE x ARO*

A szoftverbiztonság több, mint biztonsági kódokkal és becslésekkel védekezni, vagy biztonsági rések után kutatni a kódban, vagy a támadók elleni egyéb védelem. Figyelembe kell venni az emberi tényezőt is, például előfordulhat, hogy egy olyan ember ítéli meg egy rendszer biztonságát, aki nem szakértő, és ezért hibázik.

Egy másik nagyon fontos szempont azoknak az adatoknak a kezelése, amik üzleti titkoknak minősülnek. Ha a vállalat rákényszerül, hogy külső munkatársat alkalmazzon, akkor nagyon pontosan meg kell határozni, hogy mihez férhet hozzá az az ember, mert ha nyilvánosságra hoz kritikus fontosságú adatokat, azzal sokat veszíthet a cég.

A szoftver-forráskódja, valamint programkódja- és a hozzá tartozó dokumentáció a programozók szellemi alkotása. Mindezen alkotások szerzői jogával tehát a szoftver alkotója rendelkezik, tehát a szoftver létrejöttének pillanatában az alkalmazás már jogvédelem alatt áll. Természetesen vannak olyan szoftverek is, melyek teljes egészében ingyenesen használhatók, vagy egy bizonyos időre kipróbálhatók.

A szoftverek felhasználására licenc vásárlásával szerezhetünk jogot. Ha viszont a licencszerződést megszegjük, törvényt sértünk. Tehát törvényt sért:

- aki szoftvert, vagy annak dokumentációját, beleértve a programokat, alkalmazásokat, adatokat, kódokat és kézikönyveket szerzői jog tulajdonosának engedélye nélkül lemásolja vagy terjeszti,
- aki szerzői jog által védett szoftvert egyidejűleg két vagy több gépen futtat, hacsak ezt a szoftver licenc szerződése külön nem engedélyezi,
- az a szervezet, amely tudatosan vagy akaratlanul munkatársait arra ösztönzi, kötelezi, vagy számukra megengedi, hogy illegális szoftvermásolatokat készítsenek, használjanak, vagy terjesszenek,
- aki az illegális szoftvermásolást tiltó törvényt megsérti azért, mert valaki erre kéri vagy kényszeríti,
- aki szoftvert kölcsön ad úgy, hogy arról másolatot lehessen készíteni, vagy aki a kölcsönkért szoftvert lemásolja,
- aki olyan eszközöket készít, importál, vagy birtokol, amelyek lehetővé teszik a szoftver védelmét szolgáló műszaki eszközök eltávolítását, vagy ilyen eszközökkel kereskedik.

***Fogalmak a 2.5. fejezetben:*** SLE (Single Loss Expectancy), ARO (Annual Rate of Occurence), ALE (Annual Loss Expectancy).

## 2.6 Kockázatok lehetséges kezelési módjai

A kockázatok azonosítása és értékelése önmagában nem biztosítja a projekt sikerét. A projekt akkor lesz sikeres, ha a felmerült kockázatokra megfelelő módon reagálunk (és ez nem csak a biztonsági kockázatokra igaz). Egy nagyobb projektnél viszont mindenképpen figyelembe kell venni a projekt paramétereit és korlátait is (határidők, erőforrások, költségek, lehetőségek, stb.), és ezek ismeretében kell eldönteni, hogy a felmerült kockázatokat milyen módon kívánjuk kezelni.

A kockázatok csökkentésének illetve kivédésének több módja is van, de az is előfordulhat, hogy jobban megéri bevállalni és a bekövetkezés esetén kis költséggel reagálni rá, mint sok erőforrást költeni a kivédésére. A következő példán keresztül tekintsük át, hogy általában milyen kockázatkezelési lehetőségeink vannak.

Tegyük fel, hogy egy cég üzemeltet egy internetes áruházat. Ilyenkor ennek meg kell felelni a PCI DSS (Payment Card Industry Data Security Standard) szabványnak, hogy megvédje a kártyatulajdonosok adatait.

### 2.6.1 A kockázat figyelmen kívül hagyása

A vállalat dönthet úgy, hogy figyelmen kívül hagy egy bizonyos fajta kockázatot. A kockázat kezeletlenül marad. Nem javasolt módszer, mert a vállalat könnyen szembekerülhet a törvénnyel, ha a az ügyfelei adatait nem védi megfelelő mértékben. Ilyenkor nyilvánvalóan az ügyfelek sem fognak bízni az adott cégben.

### 2.6.2 A kockázat elkerülése

Mivel ez a cég fő bevételi forrása, nem célszerű alkalmazni, de adott szituációban dönthetnek úgy, hogy nem akarják tovább használni az adott szoftvert.

### 2.6.3 A kockázat csökkentése

A fejlesztőcsapat dönthet úgy, hogy különböző biztonsági technikákkal csökkentik a kockázatot. Például különféle biztonsági protokollokat használnak (SSL, TLS), vagy az adatokat titkosítják, mielőtt tárolnák.

Ezzel azt hihetnénk, hogy teljesen biztonságban vannak az adatok, de az adatok visszafejthetők, ha rosszul implementált vagy túl egyszerű a titkosítás.

### 2.6.4 A kockázat elfogadása

Ebben a helyzetben a vezetőség úgy dönt. hogy tudomást vesz a kockázatról, de annak a hatása vagy a bekövetkezési valószínűsége elenyésző. Az ilyen döntéseket pontosan kell dokumentálni, és nem szabad a későbbiekben sem figyelmen kívül hagyni. Ilyenkor általában készül valamilyen forgatókönyv az esetleges bekövetkezés esetére, „kártalanításra".

### 2.6.5 A kockázat áthelyezés

Ez általában nem a tényleges kockázat áthelyezését jelenti, hanem csak a felelősség átruházását, ettől a kockázat még ugyanúgy és ugyanott létezni fog.

Általában ez biztosítások kötésével és jól megfogalmazott jogi nyilatkozatokkal lehetséges. Habár a szoftverbiztonsági biztosítások nem mondhatók általánosnak, mégis ezek a legjobb megoldások, ha a kockázat kezelésének költségei meghaladnák az okozott károk várható összegét. Ezért van például, hogy a szoftvereket csak úgy tudjuk telepíteni, hogy elfogadjuk a végfelhasználói feltételeket, így a fejlesztő cég leveszi a felelősséget a válláról.

***Fogalmak a 2.6. fejezetben:*** kockázatcsökkentés, PCI DSS, SSL, TLS, titkosítás.

## 2.7 Lehetséges biztonsági kockázatok

Ebben a fejezetben megnézünk néhány módszert arra, hogy a támadók milyen módszereket használhatnak a „munkájuk" elvégzésére.

### 2.7.1 Buffer Overflow

A programok rossz input kezelését kihasználó feltörési technika. Lénygében azon alapszik, hogy a legtöbb program a paramétereket, vagy akár az egyszerű billentyűzetről kapott inputot is ellenőrzés nélkül, a vermen keresztül olvassa be. Tételezzük fel, hogy a beolvasás egy saját eljárásban/függvényben történik.

Itt lefoglalunk az inputnak mondjuk 100kB-ot (buffer), majd meghívjuk a beolvasó rutint. A legtöbb beolvasó rutin nem ellenőrzi a beolvasott adat hosszát, így ha példánkban több mint 100kB olvas be, attól függően, hogy melyik

irányból kezdte a feltöltést, könnyen felülírhatja az egyik visszatérési címet. Ennek hatására a függvény végrehajtásakor a vezérlés könnyen olyan memóriaterületre adódik át, amit a programozó nem akart. Ha a támadó tudja, milyen memóriaterületre jut így el, akkor elmondhatjuk, hogy a program fel lett törve, az adott program jogaival bármit megtehet az, aki ilyen módon behatol a rendszerbe. Hatékonyan csak a programok írásakor lehet ellene védekezni, ha nem használunk olyan beolvasó rutint, ami nem veszi figyelembe a buffer hosszát.

Két fajtája a buffer overflow-nak:

- Stack Overflow
- Heap Overflow

## 2.7.2 Injections

A támadók kihasználhatnak különböző biztonsági réseket. Az ilyen fajta támadások esetén egy kódrészletet küldenek a fordítónak, majd a program végrehajtja ezt az utasítást.

Ez azért lehetséges, mert nem ellenőrzik, hogy a parancs megbízható forrásból származik-e.

Fajtái:

- SQL injection
- OS Command injection
- LDAP injection
- XML injection

## 2.7.3 Broken Authentication and Session Management

Amennyiben egy felhasználó hitelesítése nem kellőképpen biztonságos, akkor lehetséges, hogy feltörjék a jelszavakat, így ellopják más emberek identitását.

**Cross-Site Scripting (XSS):** Az XSS a számítógépes sebezhetőség egy fajtája, amely tipikusan web alkalmazásoknál fordul elő: egy rosszindulatú web-felhasználó olyan kódot illeszt egy weblapra, amit más felhasználó is lát.

Például ilyen kód lehet a HTML kód vagy egy kliens oldali script. Ha egy támadó egy XSS sebezhetőséget felfedez, azt felhasználhatja arra, hogy a hozzáférési ellenőrzést kikerülje, például azzal, hogy a böngésző által kapott weblap nem az eredeti forrásból származik (de megjelenésében azonos lehet az eredetivel).

Fajtái:

- Tükrözött (non-persistent) XSS
- Tárolt (persistent) XSS
- DOM alapú XSS

## 2.7.4 Insecure Direct Object References

Ha direct hivatkozásokat használunk, akkor a támadóknak lehetőségük van kikerülni a hitelesítési eljárást, és így olyan adatokhoz férhetnek hozzá, amelyekhez nem lenne jogosultságuk.

## 2.7.5 Security Misconfiguration

Lehetséges kockázat az is, ha nem megfelelően van konfigurálva a rendszerünk, például nem használunk tűzfalat vagy egyéb eszközöket a szervereink védelme érdekében.

## 2.7.6 Sensitive Data Exposure

Sok webes alkalmazás nem megfelelően védi az adatokat, sem amikor tárolja, sem amikor szállítja/használja (titkosítás, biztonságos szállítási mechanizmus).

### 2.7.7 Missing Function Level Check

Ha a böngésző egy erőforrást szeretne igénybe venni, akkor az összes alkalmazásnak hitelesítenie kell magát (pl.: URL). Ha ez a hitelesítési folyamat hibás, akkor ezt kihasználhatják a támadók.

### 2.7.8 Cross-Site Request Forgery (CSRF)

Ez a fajta támadás, egy alkalmazásba bejelentkezett felhasználót vehet célba. Folyamata, hogy ellopják a bejelentkezés során a gépre letöltött cookie-kat és ezzel – ha túl sok információt tárol az alkalmazás benne – hozzáférhetnek a támadók bizalmas adatokhoz.

### 2.7.9 Using Known Vulnerable Components

A sebezhető komponensek library-k, framework-ök és más szoftvermodulok majdnem teljes hozzáférési jogosultsággal futnak, így ha ezek kárt okoznak, azok könnyen a teljes rendszer vesztét okozhatják.

### 2.7.10 Unvalidated Redirects and Forwards

További információ:
https://www.owasp.org/index.php/Unvalidated_Redirects_and_Forwards_Cheat_Sheet

***Fogalmak a 2.7. fejezetben:*** puffertúlcsordulás, stack overflow, heap overflow, SQL injection, OS Command injection, LDAP injection, XML injection, Broken Authentication and Session Management, Cross-Site Scripting (XSS), Non-persistent or Reflected XSS, Persistent or Stored XSS, DOM based XSS, Insecure Direct Object References, Security Misconfiguration, Sensitive Data Exposure, Missing Function Level Checks, Cross-Site Request Forgery (CSRF), Using Known Vulnerable Components, Unvalidated Redirects and Forwards.

# 3. KÖVETELMÉNYEK ÉS ALKALMAZÁSBIZTONSÁG

## 3.1 Rendszer és környezete

A követelmények fontosságát nem lehet eléggé hangsúlyozni, ugyanis ezek határozzák meg a szoftvert.

Minél kevésbé, minél lazábban definiáltak a követelmények, annál valószínűbb, hogy a szoftver fejlesztése során ötletszerű, esetleg rossz megoldások kerülnek alkalmazásra, amelyek rontják a szoftver minőségét és megemelik a fejlesztési, tesztelési és végső soron az üzemeltetési költségeket is.

A követelmény specifikációra ezért egyre nagyobb hangsúlyt fektetnek a cégek. Ugyanakkor követelmények alatt jellemzően funkcionális követelményeket értenek, és a biztonsági követelményeket egyáltalán nem, vagy csak érintőlegesen, magas szinten fogalmaznak meg. Az „A jelszavakat biztonságos módon kell kezelni" mondat – ha egyáltalán explicite szerepel a követelmények között – sokkal valószínűbb, mint a jóval pontosabb „A jelszavak beviteli mezőit maszkolni kell, a jelszavak átvitelére csak SSL titkosítású csatorna használható és a rendszer a jelszavakat hash formában tárolja." megfogalmazás. Márpedig ha a kifejlesztendő rendszer biztonságáról nem gondoskodunk megfelelően a követelmények szintjén, azt később eléggé nehéz, vagy inkább lehetetlen megvalósítani.

Amikor a rendszer követelményeiről beszélünk, meg kell különböztetnünk magát a kifejlesztendő **rendszert**, a **kontextust** és **az irreleváns környezeti elemek**et.

A **rendszer** az, amire befolyással vagyunk. A különféle követelményekkel befolyásolhatjuk például a rendszer működését, felépítését, használhatóságát vagy biztonságosságát. A **kontextus** az a környezet, amelybe a rendszerünket helyezzük. Ez nem kizárólag hardver-szoftver környezetet jelent, hanem ide tartoznak a rendszerrel kapcsolatos szerződések, jogi szabályozás, céges policyk és gyakorlatilag minden, amiből a rendszerrel szemben elvárások, követelmények fogalmazhatóak meg. **Az irreleváns környezet** pedig az, aminek nincs befolyása a rendszer követelményeire.

A rendszer és a kontextus közötti határvonal feladata meghatározni azt, hogy a rendszerünkbe mi kerül bele. Ha egy komponens a rendszer része, akkor vele szemben követelményeket fogalmazhatunk meg, ám ha a kontextusba esik, akkor máris a követelmények meghatározó tényezőjévé válik. Ha például az adatátviteli csatornát a rendszerhez kapcsoljuk, akkor azon majd módosításokat hajthatunk végre a biztonság érdekében a megfogalmazott követelmények alapján. Ám ha ez a kommunikációs eszköz a kontextus része, akkor adottak a lehetőségeink, és a biztonságot más módon kell szavatolni.


*4. ábra Biztonsági követelmények felépítése*[1]

1 http://www.opensecurityarchitecture.org/cms/definitions/it_security_requirements

A rendszer és a környezet összekapcsolására interfészeket használunk. Ezek lehetnek a kontextus által adott, nem változtatható interfészek (pl. külső modul API-ja, szabványos fájlformátum), illetve a rendszer részét képező elemek (pl. GUI) is. A rendszerünk ezeken keresztül kommunikál a külvilág adatforrásaival (a rendszer inputjának forrásai) és adatnyelőivel (a rendszer kimenetének feldolgozói). Léteznek kombinált (egyszerre forrás és nyelő) entitások is.

A kontextus és az irreleváns környezet elhatárolása szintén fontos. Minél több elemet tartalmaz ugyanis feleslegesen a kontextus, azokon keresztül annál több irreleváns és szükségtelen követelményt tudunk definiálni. Ennek már önmagában sincs jó hatása a projektre nézve, de biztonsági megfontolásokból kifejezetten káros, ha a felesleges követelmények komplexebbé, átláthatatlanabbá teszik a rendszert és elvonják a figyelmet, illetve az erőforrásokat.

A rendszert, a kontextust és az irreleváns környezetet tehát világosan el kell határolni, ez azonban nem mindig egyszerű. A „szürke zónák", amelyekben ez a két határvonal (rendszer-kontextus és kontextus-irreleváns környezet) mozoghat, általában csak a követelmények véglegesítésekor tűnnek el teljesen (ha egyáltalán eltűnnek).

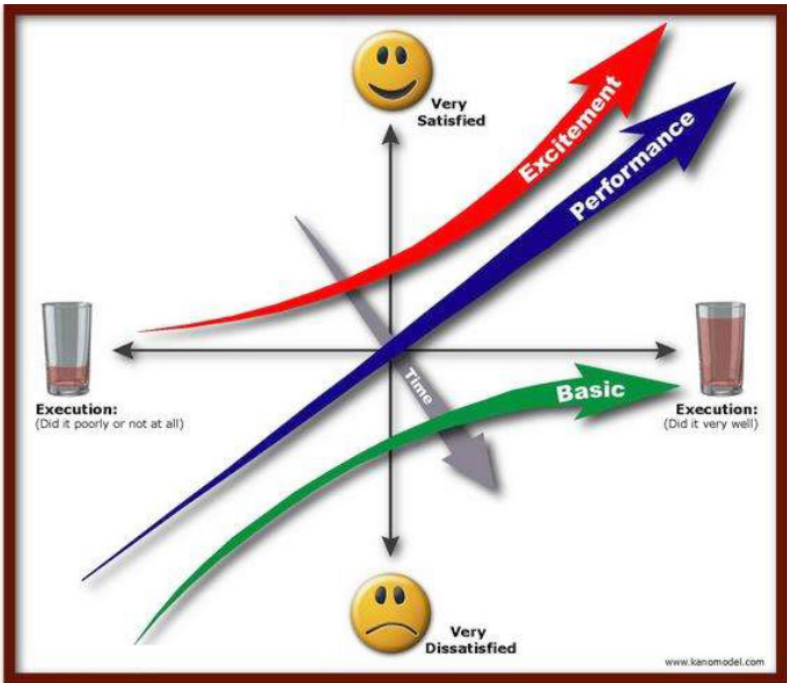
Például vegyünk egy könyveket áruló webáruházat, ahol a rendszer követelményei közé tartozik az, hogy a felhasználó egyedi azonosítóval tudjon belépni, és csak belépés után tudja használni a rendszert. A kontextus legyen egy szerver, ahonnan az alkalmazás elérhető lesz minden olyan szabályozással, amely a biztonságos online vásárláshoz szükséges. Irreveláns környezeti elem pedig ebben az esetben lehet pl. egy kevésbé nagy jelentőséggel bíró mező funkcionalitásának hosszas ecsetelése.

***Fogalmak a 3.1. fejezetben:*** kontextus, követelmény specifikáció, irreveláns környezeti elem, interfész.

## 3.2 Követelmény feltárási technikák

A követelmény feltárás az a lépés, amikor a rendszer kontextust felhasználva különféle forrásokból meghatározzuk a rendszer követelményeit. Ez a lépés különösen fontos a biztonsági követelmények tekintetében, hiszen a rendszer biztonságát tulajdonképpen itt fogjuk meghatározni (Protection Needs Elicitation).

Követelményforrás alatt értünk bármit/bárkit amit/akit felhasználunk arra, hogy a követelményeket meghatározzuk. Forrás lehet például a rendszer fejlesztésében érintett vagy arra befolyással bíró személy vagy szervezet (felhasználók, fejlesztők, tesztelők, operátorok, a megrendelő cég képviselői, stb.), valamilyen általános, szakterület- vagy cégspecifikus dokumentum (nemzetközi standardok, törvények, szerződések, régi követelmények, hibajelentések, stb.) illetve valamely működő rendszer (legacy rendszerek, a szoftver előző verziója, valamely konkurens termék, stb.).




*5. ábra Kano-modell* [2]

2 http://jaytchakarov.blogspot.hu/2011/02/table-stakes-or-just-covering-basics-of.html

A fentiek közül kiemelt figyelmet kell fordítani a humán forrásokra. Érdemes a projekt elején listát készíteni minden lehetséges érintett fontosabb adataival (név, elérhetőség, szerep a projektben, szakterület, stb.) és ezt a listát a projekt során folyamatosan karbantartani. Az érintett feleket érdekelté kell tenni a projektben, aktívan be kell vonni őket a követelmények feltárásába, ellenőrzésébe, megvitatásába. Időről-időre tájékoztatni kell őket a feltárás állásáról, az aktuális követelményekről.

A Kano-modell szerint háromféle követelményt különíthetünk el: **tudatalatti** (dissatisfier), **tudatos** (satisfier) és örömszerző (delighter). A tudatalatti olyan evidens követelmény, amit ha megvalósul észre sem vesszük, azonban ha hiányzik, az gyakorlatilag használhatatlanná teszi a rendszert. A tudatos követelmények adják a szoftver „gerincét". Ha egy-egy ilyen hiányzik, akkor a rendszer nem használható, nem tölti be a funkcióját, a megfelelő megvalósítása viszont érzékelhető és beazonosítható. Az örömszerző követelmény az amit nem hiányolunk a rendszerből, a megvalósítása viszont határozottan emeli a szoftver színvonalát, használhatóságát.

Egy-egy adott követelmény nem biztos, hogy mindig ugyanabba a kategóriába esik, idővel és a körülmények változásával egy örömszerző már természetes, tudatos követelmény lesz, később pedig evidenssé válik. Gondoljunk csak egy cég raktárnyilvántartó szoftverére. Amíg egy kis garázscégről van szó az adatokhoz való hozzáférés korlátozása (azonosítás, jogosultságok kezelése) szóba sem kerül, egy ilyen funkció egyértelműen örömszerző kategória.

A cég növekedésével viszont előbb-utóbb szükség lesz valamiféle korlátozásra, tehát ezek a követelmények tudatossá válnak, egy nagyvállalat rendszerei pedig elképzelhetetlenek valamiféle felhasználói azonosítás és jogosultságkezelés nélkül.

A követelmények meghatározását különféle technikák segíthetik. Ezek közül egyik sem abszolút jobb vagy rosszabb a másiknál, de a körülményektől függően (a követelmények Kano besorolása, projekt erőforrásai, követelménymérnök tapasztalata, kockázatfelmérés eredménye, egyéni emberi és szervezeti tényezők, szakterület, elvárt részletesség, stb.) jobban vagy kevésbé éri meg alkalmazni őket. A technikákat több csoportba sorolhatjuk.

## 3.2.1 Kikérdezés technikák

A technikák első csoportját a **kikérdezéses** (survey) technikák alkotják, mint például a személyes **megbeszélések** (interview) vagy a **kérdőívezés** (questionnaire). Ezek általában időigényes technikák: az interjúk önmagukban, a kérdőívezésnél pedig főként az előkészület, a helyes kérdések megfogalmazása. A személyes megbeszélések inkább a tudatos, a kérdőívek a tudatalatti követelmények meghatározására előnyösek. A kérdőívek előnye, hogy nagyon nagy számú, összességében sok szempontot magában foglaló válaszokat kapunk a kérdéseinkre. A technikák alkalmazásához nem árt némi tapasztalat, interjúk esetében pedig megfelelő kommunikációs képességek (a kérdezettek részéről is).

Mindkét módszer kifejezetten alkalmas lehet biztonsági követelmények összegyűjtésére, és ekkor fontosak az üzleti, projekt és alkalmazás kockázatára vonatkozó kérdések. A kérdések összeállításakor a szoftverbiztonsági profil és a biztonságos tervezés szabályainak figyeleme vételével akár közvetlenül biztonsági követelményekké konvertálható válaszokat is kaphatunk. Fontos továbbá a nyílt kérdések használata, különösen a kérdőíveken, amikor is sokkal kisebb az interakció lehetősége.

## 3.2.2 Kreatív technikák

A következő csoport a kreatív (creativity) technikák csoportja. Ezek közé sorolhatók a **brainstorming**, **nézőpontváltás** (change of perspective), **analógia** (analogy) technikák. Jellemzően alkalmasak a örömszerző követelmények felsorolására, kötetlen formátumúak és időhatékonyak, alkalmazásukhoz nincs szükség előképzettségre, ugyanakkor nem eredményeznek részletes követelményeket. A legjobb hatásfokot akkor érhetjük el, ha sokféle érintett vesz részt benne.

A brainstormingot érdemes kisebb csoportokban, időkeretekkel alkalmazni, ahol nem a felmerült gondolatok megvitatása, csupán az összegyűjtése lesz a cél, azok kiértékelése később történik meg. Biztonsági követelmények összegyűjtéséhez különösen hasznos lehet a brainstorming paradox-nak nevezett változat, ahol a „mit szeretnénk" helyett a „mit szeretnénk elkerülni" kérdés a központi téma. A nézőpontváltás technikája rákényszeríti a résztvevőket, hogy más-más szempontból vizsgálják a rendszert, ezen szempontok egyike lehet a biztonság, vagy más felosztásban a biztonsági követelményeket lehet fejlesztői, felhasználói, menedzsment vagy üzemeltetői szemszögből körbejárni. Az analógia technika lényege, hogy egy másik tudományterület problémáival és megoldásaival vonunk párhuzamot. Ez analitikus és elvont gondolkodást igényel, és gyakran hosszabb időt.

### 3.2.3 Dokumentum központú technikák

A dokumentum központú (document-centric) technikák korábban már valamilyen formában megfogalmazott, leginkább tudatalatti követelmények meghatározására jók. Ilyenek a **rendszerbányászat** (system archeology), **nézőpont szerinti olvasás** (perspective-based reading), újrafelhasználás (reuse) és a **policy decomposition**. Alkalmazásukhoz szükséges a korábbi követelmények valamilyen gyűjteménye (amely lehet akár egy működő rendszer is), és sok idő.

A rendszerbányászat segítségével már létező dokumentációból vagy implementációból nyerhetünk ki szisztematikus módon akár részletes követelményeket is. A perspective-based reading technika a követelmények szelektálásában segít, alkalmas például csak a biztonsági követelmények kinyerésére. Az újrahasznosítás az előzőekhez képest időhatékony lehet, hiszen változtatás nélkül veszi át a követelményeket. A policy decomposition segítségével magasszintű módszertani dokumentációból (ipari standardok, törvényi előírások, szerződések) vagyunk képesek részletes követelmények szisztematikus előállítására. Ez a módszer különösképpen fontos a biztonsági követelmények tekintetében, ahol a security policy dokumentum nagyon jó kiindulópont.

### 3.2.4 Megfigyelés technikák

A megfigyelési (observation) technikák lényege, hogy a meglévő rendszert működés közben vizsgáljuk, és ennek segítségével nyerjük ki a rendszer követelményeit. Lehetséges fajtái a **field observation** és **apprenticing**. A módszer meglehetősen időigényes, de ha a rendszer ismerői, szakértői valamilyen oknál fogva nem érnek rá, ez maradhat az egyetlen megoldás. Leginkább a gyakran használt tudatalatti követelmények kinyeréséhez használható. A biztonsági követelményeknek csak egyes fajtáit lehet általa meghatározni.

Field observation esetén a rendszert egy „normál" felhasználó kezeli – általában a felhasználó telephelyén –, a követelmény-mérnök csak passzív megfigyelő, aki a lépéseket dokumentálja (persze némi interakció, egyes lépéseket tisztázó kérdések feltehetőek). A folyamat rögzítése (audio/video) hasznos segítség lehet. Az apprenticing technika esetén a követelmény-mérnök megtanulja használni a rendszert, miközben szakterületi ismeretekre is szert tesz. Ennél a technikánál is szükség van olyan szakemberekre, akikhez igényesetén kérdéseket lehet intézni, de őket ez a feladat nem szabad, hogy lekösse.

### 3.2.5 Egyéb technikák

Meg kell még említeni azokat a technikákat, amik bár önmagukban nem követelményfeltárásra valók, nagyon hasznos kiegészítői lehetnek a fentieknek, ellensúlyozva azok hiányosságait/hátrányait.

A **tudástérkép** fogalmakat, ötleteket, gondolatokat és ezek közötti kapcsolatokat vizualizál hasonlóképpen ahhoz, ahogyan az emberi agy működik, ezért könnyen áttekinthető és megjegyezhető. Alkalmas lehet például a biztonsági szempontok csoportosítására, lebontására.

A **workshop** egy interaktív megbeszélés, ahol a résztvevők egy-egy témát járnak körül, ötleteket vethetnek fel és beszélhetnek meg viszonylag kötetlen formában.

A **Class-Responsibility-Collaboration** (CRC) kártyák inkább az alacsonyabb szintű követelmények tervezésénél lehetnek hasznosak. Nagyon közel állnak az objektumorientált gondolkodásmódhoz, entitásokat, azok tulajdonságait és más entitásokkal való kapcsolataikat lehet viszonylag egyszerűen kidolgozni a segítségükkel.

A **use case modelling** egy széles körben elterjedt technika, ahol a rendszer „szokásos" használati eseteit gyűjtjük össze, ezeknek a lépéseit írjuk le, illetve a lehetséges használók és használati módok kapcsolatait tudjuk ábrázolni. Biztonsági szempontból fontos a lehetséges használók és használati módok beazonosítása, melyből úgynevezett subject/object mátrixot állíthatunk elő. Ez segíthet a megfelelő jogosultságok rendszerezésében, megtervezésében, illetve az úgynevezett misuse case-ek definiálásában. Ez utóbbival az akaratlanul hibás, vagy szándékosan kártékony használatot próbáljuk modellezni.

A **kép- és/vagy hangrögzítés** hasznos kiegészítője lehet számos technikának, segítségével kiküszöbölhető az egyes tevékenységek többszöri megismétlése, a gyors folyamatok feltárását segítheti, csökkenti a résztvevő elérhetetlenségéből adódó kockázatokat. Leginkább (de nem kizárólag) a megfigyelési technikákkal együtt használható.

A **prototípus-készítés** az iteratív fejlesztések szinte természetes velejárója. A prototípus segít megérteni és kiértékelni a meglévő, illetve kialakítani a még hiányzó, akár részletes követelményeket.

Az **adat-osztályozás** különösen fontos biztonsági szempontból. Ennek során a rendszerben megjelenő strukturált (pl. adatbázis) és nem strukturált (pl. video, e-mail, szöveg) adatokat tudjuk bizalmassági, sérthetetlenségi és ren-

delkezésre állási (confidentality, integrity, availability – CIA) szempontokból felcímkézni (pl. publikus, közepesen sérülékeny, mindenképpen szükséges). Meg kell határozni az adat tulajdonosát és a hozzáférési jogosultságokat. Az adat teljes életciklusának modellezésével további biztonsági követelményeket határozhatunk meg (pl. adatátviteli csatorna titkosítása, redundáns tárolás).

A biztonsági követelmények meghatározása egy speciális folyamat, amely Protection Needs Elicitation (PNE)-ként is ismert. Ez a védendő elemek meghatározásával kezdődik. Az Information Assurance Technical Framework (IATF)a következő lépésekből álló módszert definiálja a hardver/szoftver biztonsági követelmények meghatározására: a megrendelővel együttműködve modellezzük a rendszert információ-menedzsment szempontból, határozzuk meg a minimális jogosultságokkal rendelkező alkalmazásokat [NOTE: least privilege], végezzünk fenyegetettségi kockázatanalízist, priorizáljunk a megrendelő igényei szerint, határozzuk meg az információbiztonsági irányelveket [NOTE: information protection policy], végül szerezzük meg a megrendelő jóváhagyását. Ezen folyamat egyes lépéseit az előző technikák kombinálásával tudjuk elvégezni.

***Fogalmak a 3.2. fejezetben:*** Kano modell, tudatalatti követelmény, tudatos követelmény, örömszerző követelmény, kikérdezés technikák, kreatív technikák, dokumentum központú technikák, megfigyelés technikák.

## 3.3 Dokumentáció természetes nyelven

Egy rendszer követelményeit három főbb szempontból lehet dokumentálni: **adat**, **működési** és **viselkedési** szempontokból. Az adat perspektíva a rendszer statikus nézete. A rendszerben előforduló adatok, információ struktúráját, felépítését, illetve a rendszer használatának és a kontextussal fennálló függőségeinek statikus, architekturális vetületét írja le. A működési nézet egy dinamikus perspektíva, ami a kontextusból érkező adatok információtartalmát, a rendszeren belüli adatfolyamokat, adatfeldolgozási lépéseket, útvonalakat, a környezetnek visszaadott adatokat írja le. A viselkedési perspektíva pedig azt ábrázolja, hogy a rendszer hogyan épül be a kontextusba, hogyan reagál a külvilágból érkező „ingerekre" és hogyan hathat a környezetére.

A fenti nézeteket alapvetően kétféleképpen dokumentálhatjuk: **természetes nyelv**en és **modell**ekkel. A természetes nyelvi dokumentáció az, aminek hangzik: a követelmények szöveges leírása. Az ilyesfajta dokumentáció legnagyobb előnye, hogy bárki számára olvasható (új jelölésrendszer megtanulása nélkül), a kifejezőereje nagy, bármilyen fajta követelményt világosan le lehet dokumentálni vele. Hátránya, hogy a különféle perspektívák gyakran keverednek az ilyen dokumentációkban, és a módosítás, javítás meglehetősen nehéz és komplex feladat lehet. A modellekkel történő dokumentáció ezzel szemben a jelölésrendszert ismerők számára sokkal egyértelműbb, érthetőbb, nyelv független, az egyes perspektívák jól elkülöníthetőek. Hátránya viszont, hogy az egyes modellek kifejezőereje általában egyetlen perspektívára korlátozódik, és speciális helyzeteket nem, vagy csak nagyon körülményesen lehet dokumentálni. Ráadásul a modelleket meg kell ismerni, ami egy-egy speciális szereplőnek nem probléma (sőt, kifejezetten követelmény), ugyanakkor nem várható el mindenkitől (a megrendelő például nem fogja a UML osztálydiagramok jelölésrendszerét megtanulni).

A kétféle dokumentáció hátrányai viszonylag egyszerűen kiküszöbölhetők, ha mindkettőt egyszerre használjuk. A modellek mellé például természetes nyelvi megjegyzéseket, kiegészítéseket tehetünk, míg egy természetes nyelvi követelmény-leírásba ábrákat szúrhatunk, hogy egyértelműsítsük a leírást.

Természetes nyelvi dokumentációnál nagyon fontos a dokumentáció formátuma. Érdemes valamilyen szabványból kiindulni (IEEE-830, RUP, V-model) és azt testre szabni.

A követelmények természetes nyelven történő dokumentációja és felhasználása során a résztvevők tudása, háttérismeretei és személyes tapasztalata két ponton is jelentősen befolyásolhatja a folyamatot. Egyrészt a valós követelmények leírásánál, dokumentálásánál, valamint a leírt követelmények értelmezésénél, felhasználásánál. Így a már eleve pontatlanul lejegyzett valós követelmények szintén pontatlan értelmezésével az eredetileg elképzelt követelményeknek meg nem felelő munkatermékeket fogunk előállítani.

A pontatlanságokat előidéző „transzformációs hatások" nagy része viszont kiküszöbölhető, ha odafigyelünk rájuk. A nevesítés például az a folyamat, amikor egy, a dokumentáló számára egyértelmű, de valójában komplex cselekvésnek nevet adunk, és a dokumentáció során így hivatkozunk rá. Ez a cselekvés több részletét elrejtheti. Ennek feloldása lehet az, ha elhagyjuk a nevesítést és eleve a cselekvést fogalmazzuk meg, vagy ha nevesítést használunk, akkor minden nevesített cselekvést külön kirészletezünk.

A dokumentációban lévő főneveket (mint cselekvés alanyát, tárgyát) gyakran megfelelő referenciák nélkül használjuk. Ilyenkor gondot jelenthet beazonosítani, hogy a több hasonló elem közül melyikre vonatkozik. A „felhasználó" például jelentheti az összes, a bejelentkezett, de a kiválasztott felhasználókat is. A főneveket tehát ahol csak lehet,

lássuk el a megfelelő jelzőkkel. Szintén pontatlanságot okozhat, ha ok nélkül vagy tévesen használjuk az univerzális kvantorokat. Az „összes”, „mindig”, „semelyik”, „soha” és hasonló szavaknál mindig végig kell gondolni, hogy valóban jogosak-e, nincs-e bármilyen kivétel. Mert ha a felhasználó valóban összes személyes adatát jól láthatóan megjelenítjük a megfelelő oldalon, akkor a jelszó maszkolatlan megjelenése bizony komoly biztonsági kockázatot rejt.

További hiba szokott lenni, amikor feltételek kombinációival nem fedjük le az összes lehetséges esetet. Azt megadjuk, hogy adott feltételek mellett mik az elvárásaink, de azt már nem, hogy a feltételek nem teljesülése esetén mi lenne a követelmény. Előfordul még az a szituáció is, amikor magát a cselekvést nem teljesen pontosan specifikáljuk. Kihagyjuk a cselekvés tárgyát vagy alanyát. Mivel erre nagyobb esély van passzív megfogalmazás során, javasolt a történés helyett valóban cselekvést leírni.

Hasznos lehet úgynevezett követelménysablonok használata. Egy-egy ilyen minta megadja a követelmények leírására használt mondatok formátumát és tartalmi elemeit.

| REQ# | Követelmény | Komment | Prioritás | Dátum | Felülvizsgálta |
|---|---|---|---|---|---|
| MOD_01 | A rendszer a jelszót kódolja | A kódolás MD5 is lehet. | 1 | 3/25/2014 | Teszt Elek |
| MOD_02 | Belépéskor felhasználónév-jelszó páros ellenőrzése | – | 1 | 3/25/2014 | Teszt Elek |

Ezt követve a dokumentációnak egyrészt egységes formátuma lesz, másrészt az előre rögzített forma segít egyes transzformációs hatások kiszűrésében. Egy ilyen template elemei a következők. Először is szükség van egy olyan határozóra, ami a követelmény kötelezőségét mutatja, és egyértelműen beazonosítja pl. a kötelező, opcionális és később megvalósítandó követelményeket. Ez a határozó esetleg utalhat a kötelezettség forrására is (pl. „törvény szerint”). Ez után jöhet a követelmény „magja”, maga a cselekmény vagy folyamat leírása. Ez még kiegészül a folyamatnak a rendszerhez való viszonyával, ami azt fejezi ki, hogy a rendszer milyen módon biztosítja a követelményt. Ezt megteheti magától önálló tevékenységként, felhasználónak nyújtott szolgáltatásként felhasználói aktivitásra, vagy a környező rendszerek számára valamely interfészen keresztül nyújtott szolgáltatásként.

E három mód leírását a mondatok szerkezetében, a szóhasználatban is világosan el kell különíteni. A következő lépés a folyamat hiányzó elemeinek megadása, melyeknek szintén helyet kell hagyni a mondatszerkezetben. Végül, de nem utolsósorban, amennyiben a leírt cselekvésnek logikai vagy időbeli feltételei vannak, ezeket is le kell írni, így a mintában ezek számára is egyértelmű helyet kell biztosítani.

A fenti elven megalkotott minta használata egységesebb és egyértelműbb követelményleírást tesz lehetővé.

***Fogalmak a 3.3. fejezetben:*** adat perspektíva, működési perspektíva, viselkedési perspektíva, nevesítés, követelmény sablon.

## 3.4 Dokumentáció modellekkel

A modellezés nagy előnye, hogy az ábrák könnyebben értelmezhetőek és megjegyezhetőek, mint az írott szöveg. A modelleknek szigorú szintaktikai és szemantikai szabályai vannak, vagyis sokkal egyértelműbbek, mint a természetes nyelven írt szövegek. Ráadásul az absztrakció szintje a modell által adott, sokkal kevésbé variálható, mint természetes nyelven.

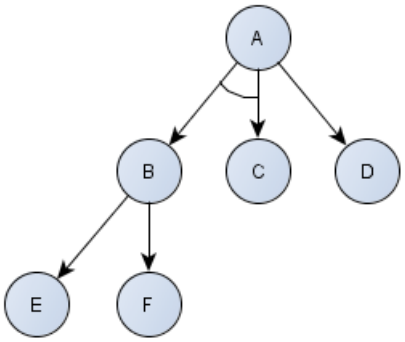
A modell a létező vagy létrehozandó valóság egy absztrakt modellje.

Tulajdonságai:

- **Reprezentációs tulajdonság**: a modell a valóság leképezése. A szükséges mértékben képes ábrázolni a létező vagy elképzelt valóságot.
- **Redukciós tulajdonság**: a modellek nem a teljes valóságot, hanem annak csak egy részét ábrázolják. Ez történhet a valóság egy részhalmazának ábrázolásával vagy a teljes valóság kevésbé részletes ábrázolásával.
- **Gyakorlatias tulajdonság**: a modelleket mindig valamilyen célból készítjük, és ideális esetben minden olyan információt tartalmaznak, ami a cél eléréséhez szükséges, és semmilyen más információt nem tartalmaznak.

### 3.4.1 ÉS/VAGY fák

A megfelelő modellekkel sokféle követelményt leírhatunk. Magas szinten például az ÉS/VAGY fák alkalmasak a fejlesztésben érdekelt felek céljainak megfogalmazására, illetve kirészletezésére, lebontására. Ezek a célok a rendszerrel szemben támasztott elvárások, jellemzők lehetnek. Jellemzőjük, hogy először magas szinten általános célok lesznek megfogalmazva (pl. legyen biztonságos az adatkezelés), és ezeket kell rész-célokra bontani mindaddig, amíg kézzelfogható, ellenőrizhető követelményekhez nem jutunk.

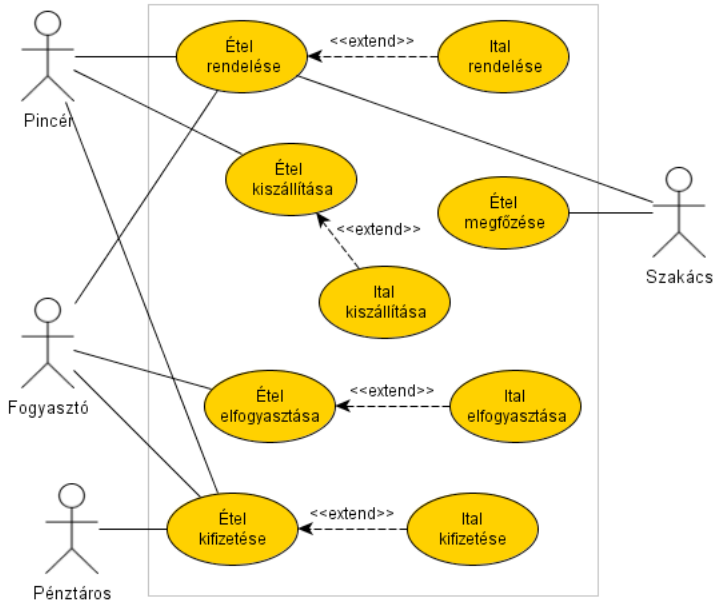



*6. ábra És/Vagy fák*

A rész-célokra bontás alapvetően kétféle módon történhet: a cél eléréséhez vagy az összes (ÉS) szükséges, vagy elegendő csak az egyik (VAGY). A célok ilyen függőségeit, lebontását lehet ÉS/VAGY fákkal ábrázolni, ami szemléletesen és könnyen követhető módon mutatja a célok függőségét és teljesíthetőségét.

### 3.4.2 Használati eset modellezés

Használati eset modellezés segítségével a rendszer (akár hibás, vagy szándékosan téves) használatának tipikus és alternatív folyamatait írhatjuk le valamely felhasználó vagy másik rendszer szemszögéből. Maga a használati eset egy szövegesen leírt lépéssorozat, meta-információkkal (pl. azonosító, név, prioritás, hivatkozások) kiegészítve. Ezen információk között megadhatóak különféle minőségre, biztonságra vonatkozó követelmények is (pl. egy külön, biztonsági követelményeket leíró dokumentum alkalmazandó elemeire való hivatkozással).




*7. ábra Példa használati eset diagramra*

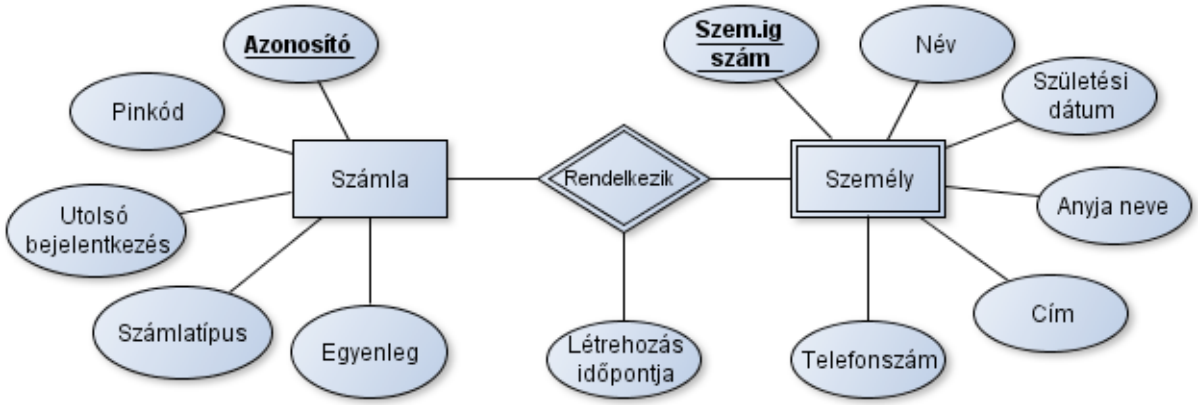
Egyes használati esetek magukban foglalhatnak vagy feltételek teljesülése esetén kibővíthetnek más eseteket, illetve több szereplővel (actor) is kapcsolatban lehetnek. Ezeknek a kapcsolatoknak az ábrázolására használati-eset diagramokat alkalmazhatunk.

## 3.4.3 Rendszer ábrázolása működési és viselkedési perspektívákkal

A rendszer követelményeit többféle nézőpontból is leírhatjuk, de a leggyakoribbak az adat, a működési és a viselkedési perspektívák. Ezek a perspektívák a modell különböző vetületei lesznek, tehát nem függetlenek egymástól.

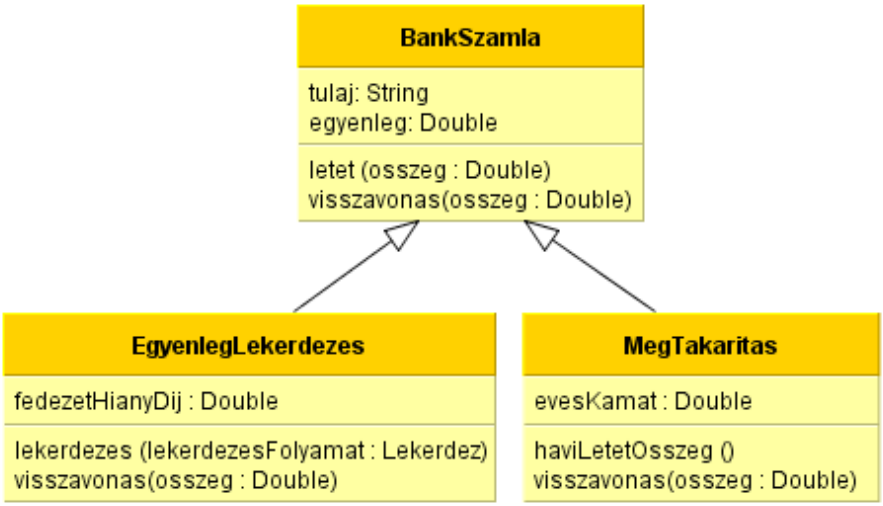
### Adat perspektíva

Az adat perspektíva a rendszerben tárolt adatok szerkezetét, struktúráját, és a rendszer használatának statikus, strukturális vetületét ábrázolhatja. Ábrázolására leggyakrabban az egyed-kapcsolat és a UML osztály diagramokat szokás használni.




*8. ábra Példa egyed-kapcsolat diagramra*

Mindkettő alkalmas a rendszerben használt entitások típusainak, azok jellemzőinek és a közöttük lévő kapcsolatok ábrázolására. A UML osztálydiagramokon a kapcsolatok valamivel részletesebben ábrázolhatók, és metódusok megadásával a működési és viselkedési nézethez is kapcsolódási pontot nyújtanak.

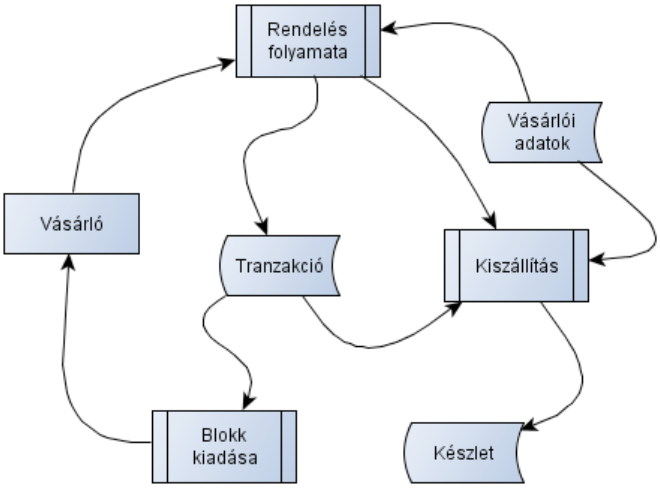



*9. ábra Bankszámla tranzakciók osztálydiagramja, leegyszerűsített példa*

Adatbiztonság szempontjából fontos nézet, hiszen tételesen tartalmazza az összes lehetséges adattípust.

## Működési perspektíva

A működési perspektíva azt írja le, hogy a rendszer a bemenő adatokból hogyan, milyen feldolgozási útvonalakon keresztül készíti el a kimenő adatokat. Ábrázolni adatfolyam diagramok vagy UML activity diagramok segítségével szokás. Az adatfolyam diagramokon a külső adatforrásokat és -nyelőket, a belső adattárolókat, az adatfeldolgozó egységeket és az ezek közötti adatáramlást lehet szemléltetni.

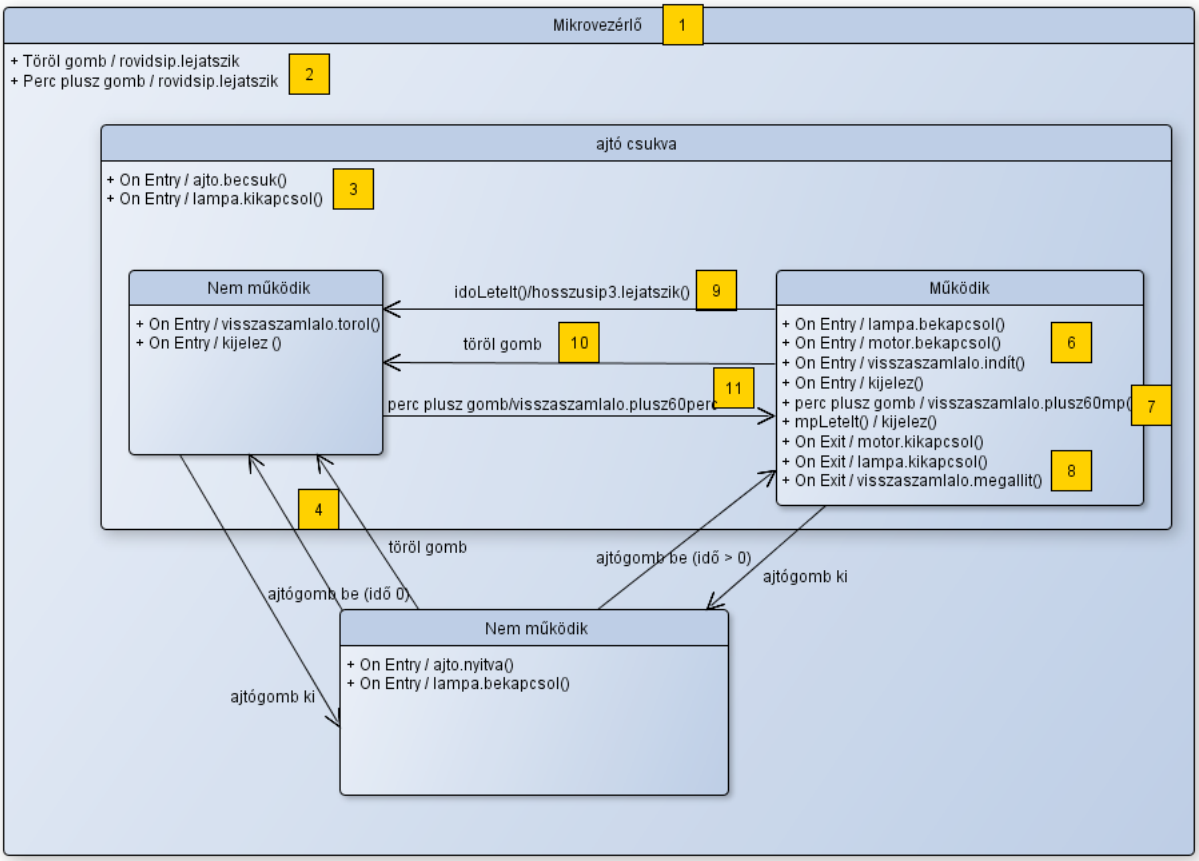



*10. ábra Adatfolyam diagram termék rendeléséről*

A UML activity diagramok ezzel szemben inkább az adatfeldolgozás vezérlési folyamatának leírásáért felelnek. Ezen a program adatfeldolgozó egységei és a közöttük áramló adatokon túl megjelenhetnek döntési pontok is. Ez utóbbiakkal alternatív vezérlési útvonalakat is ábrázolhatunk egy diagramon, valamint lehetőség van tevékenységek párhuzamos végrehajtásának ábrázolására is.

## Viselkedési perspektíva

A viselkedési perspektíva azt mondja meg, hogy a rendszer hogyan reagál a külvilág „ingereire”. Ezt a viselkedést a rendszer lehetséges állapotaival és az azok közötti átmenetekkel, vagyis automatákkal írhatjuk le. Állapotátmenet diagramot vagy UML state chart-ot használhatunk. Mindkettőben ábrázolhatjuk a rendszer állapotait, az állapotok közötti átmeneteket, az átmeneteket kiváltó eseményeket, illetve az átmenetek által előállított kimeneti adatokat. Lehetőség van hierarchikus (amikor egy állapot egy újabb automatát reprezentál) és párhuzamosan működő automaták definiálására. UML diagramon a hierarchikus ábrázolás kiegészíthető belépési, illetve kilépési pontokkal, amik egyértelműbbé teszik az automata működését.




*11. ábra Példa állapotátmenet diagramra*

Kritikus szoftvereknél ezeket a modelleket gyakran használják fel a rendszer elméleti helyességének bizonyítására.

***Fogalmak a 3.4. fejezetben:*** reprezentációs tulajdonság, redukciós tulajdonság, gyakorlatias tulajdonság, ÉS/VAGY fák, használati eset, használati eset modellezés, adat perspektíva, viselkedési perspektíva, működési perspektíva, egyed-kapcsolat diagram, UML osztály diagram, adatfolyam diagram, UML activity diagram, állapotátmenet diagram, UML state chart.

## 3.5 Validáció és egyeztetés

A követelmények meghatározásának fontos lépése a követelmények validációja. Ez az a folyamat, amikor a dokumentált követelményeket ellenőrizzük, hibákat keresünk bennük és ezeket javítjuk. Olyan kérdésekre keressük a válaszokat, mint „Megfelelő minőségűek a követelmények?”, vagy „Használható alapját képezik-e a további tevékenységeknek?”. A validáció tulajdonképpen egy korai tesztelési folyamat, melyben statikus tesztelést, review-kat tudunk alkalmazni (ezekről később lesz szó részletesebben). Korai folyamatként jelentős mértékben csökkenti a projekt fejlesztési költségeit azáltal, hogy az esetleges hibákat már az elején felfedezzük, így azokból később nem lesz hibás alapokra épülő hibás, javítandó vagy eldobandó munkatermék.

Fontos, hogy a review-k során előre rögzítsük, hogy a követelmények milyen kritériumoknak kell, hogy megfeleljenek. Tükrözniük kell például az érdekelt felek elképzeléseit, egyértelműeknek és teljesnek kell lenniük (egyenként és összességében is), nem lehetnek ellentmondásban egymással, és minden résztvevőnek el kell fogadnia őket.

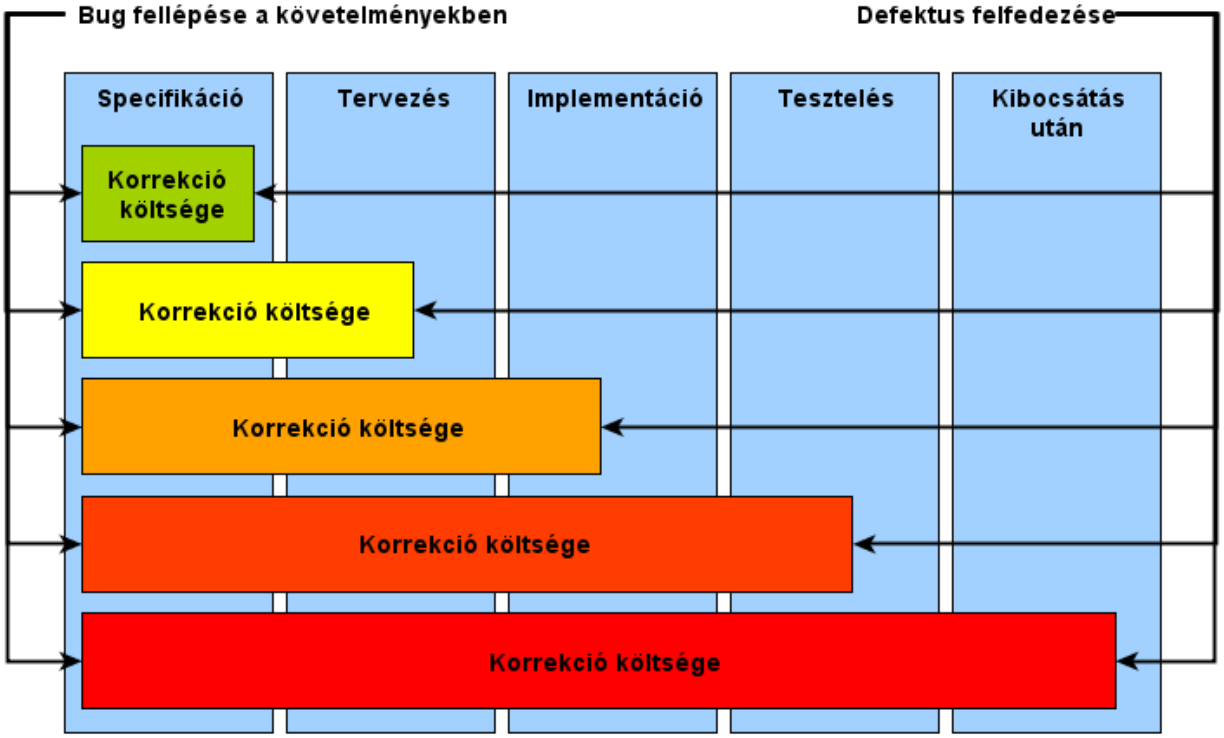
A követelményekkel szemben támasztott minőségi elvárásokat három csoportra oszthatjuk:

– tartalmi (content),
– formai (documentation) és
– elfogadottsági (agreement) csoportokra.

A tartalmi követelmények ellenőrzése során a következő típusú hibákra kell odafigyelni. Fontos a követelmények teljessége mind egyedileg, mind a követelmények összességét tekintve. Az egyes követelményekhez meg kell adni, hogy mi a forrásuk, honnan származnak. Fontos, hogy helyesek és konzisztensek legyenek, megfeleljenek a dokumentum céljainak. Meg kell vizsgálni, hogy az egyes követelmények valóban szükségesek-e, illetve implementáció után ellenőrizhető lesz-e az implementáció a követelmény alapján. A formai, dokumentációs elvárások tekintetében az előre definiált sablonok, modellek helyes használatát, a leírtak nyelvi értelmezhetőségét, egyértelműségét tudjuk ellenőrizni. Ezzel kiküszöbölhetőek olyan hibák, mint a későbbi téves interpretáció vagy rejtett követelmények kihagyása.

Az elfogadottsági kritériumok alapján pedig azt vizsgálhatjuk, hogy az egyes követelmények valóban minden résztvevő számára megfelelőek-e, az egyes érdekütközések megfelelően fel lettek-e oldva.

Validáció során nagyon fontos, hogy a megfelelő (pl. független, külső) embereket vonjuk be a folyamatba, az ellenőrzési és javítási tevékenységeket különítsük el. A validációt többféle szempontból, többféle szerep szerint végezzük el (felhasználó, üzemeltető, biztonsági szakember). És nem szabad figyelmen kívül hagyni, hogy a fejlesztési folyamat során a követelmények változhatnak, tehát a validáció nem egyszeri, hanem folyamatos tevékenység.




*12. ábra Validáció előnyei*

A validáció során gyakran fogunk konfliktushelyzetekkel szembesülni. Konfliktusok jelentkezhetnek a követelmények vagy az érdekelt felek között is. A konfliktus feloldásához fontos megállapítani az ellentét típusát. Ez adódhat abból, hogy az érdekelt feleknek különböző céljaik vannak (pl. a felhasználó a loggolásban a teljesítmény csökkenését látja, az operátor pedig a hibafelderítéshez szükséges információ előállítását), másként értékelik az egyes lehetőségeket. Az érdekelt felek személyes vagy szervezeti viszonya (pl. főnök-beosztott) is eredményezhet konfliktusokat, illetve a konfliktus tárgyáról való hiányos ismereteknek is lehet ilyen eredménye. A konfliktus típusa általában nem sorolható be egyértelműen egyetlen kategóriába.

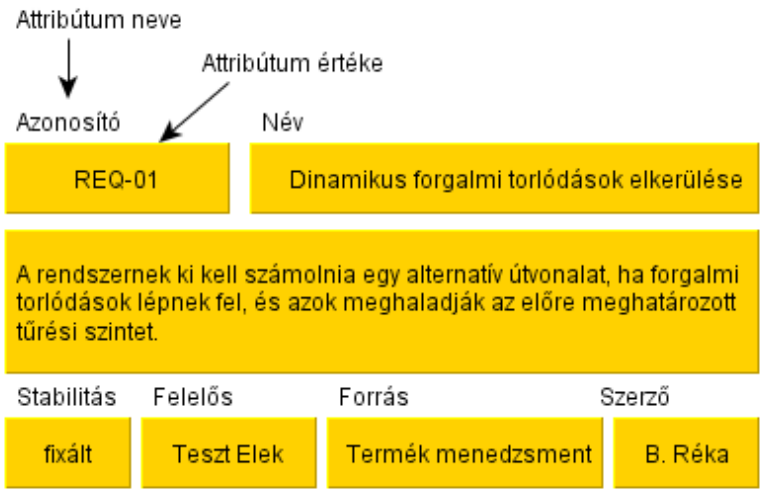
A konfliktusok feloldására többféle módszer létezik. Az egyik fél meggyőzheti a másikat a saját verziójáról, vagy többségi döntéssel eldönthetik a kérdést. Ha lehetséges, készíthetnek egy harmadik, mindenkinek megfelelő kompromisszumos változatot, vagy egyszerre megvalósíthatják mindkét alternatívát. Analitikusabb módszer, ha bizonyos szempontok szerint kiértékelik a lehetséges megoldásokat (jó/nem jó/semleges címkék, pontrendszer). Ha sehogyan nincs megoldás, a legfőbb döntéshozó fog eredményt hirdetni.

A validáció és egyeztetés folyamatába mindenképpen be kell vonni biztonsági szakembereket, akik a szoftver biztonsági követelményeiért lesznek felelősek. Továbbá a döntési pozícióban lévő embereknek meg kell értenie, hogy a biztonsági követelmények a fontosabbak közé tartoznak.

***Fogalmak a 3.5. fejezetben:*** validálás, statikus tesztelés, review, tartalmai követelmények, formai követelmények, elfogadottsági követelmények, naplózás (loggolás), konfliktus.

## 3.6 Követelmény-menedzsment

A megfelelő követelmény-menedzsment gondoskodik a követelmények kezeléséről. Ez magában foglalja a követelménygyűjtés kezdeti időszakát, a követelmények kidolgozását és későbbi változtatását is. Ezeknek a folyamatoknak átláthatóaknak kell lenniük és megfelelő módon dokumentálni kell őket. Ehhez elengedhetetlen, hogy a követelmények ne csak szimplán leírva legyenek, hanem megfelelő meta-információval is el legyenek látva. Ilyen meta-adat az egyes követelményekhez rendelt egyedi azonosító, név, verziószám, a követelmény forrása, prioritása, típusa, feltárás közbeni állapota, stb.




*13. ábra Példa követelmény sémára*

A követelmények folyamatosan változhatnak. Ennek oka lehet például a hibák javítása, a kontextus változása, új igények, stb. Az egyszer már felállított követelményeket viszont nem lehet csak úgy átírni, ezt a folyamatot kontrollálja a változás-menedzsment. Egy változtatás egy változtatási igény beadásával kezdődik, amit a Change Control Board fog elbírálni. Az igénynek tartalmaznia kell a konkrét változás leírásán túl indoklást, egyedi azonosítót, dátumot, igénylőt, stb. A board-ot úgy kell összeállítani, hogy a tagjai a fejlesztési-üzemeltetési folyamat minden lépését és szempontját reprezentálják. A board feladata többek között a kérelem osztályozása (javítás, új igény, gyorsfix), priorizálása, a változás erőforrásigényének megbecslése, a mehet/nem mehet/mikor mehet döntés meghozatala és pozitív döntés esetén a változás projektesítése. Az erőforrásigény megbecsléséhez nagyon fontosak a traceability információk, hiszen ezek segítségével lehet megmondani, hogy a változtatásnak mekkora lesz a hatása, várhatóan mekkora erőforrás kell majd a végigviteléhez (követelményektől fejlesztésen át a tesztelésig).

Traceabilitynek azt a meta-információt nevezzük, ami kapcsolatokat, összefüggéseket definiál az egyes követelmények között, megadja az egyes követelmények eredetét (azt a forrást, ami vagy aki a követelményt meghatározta), és megmondja, hogy az egyes követelményekre pontosan mely további munkatermékek (modulok, tesztesetek) épültek. Ezt az információt az egyes követelményeknél rögzíteni kell, mert jelentősen megkönnyíti a változás hatásának megbecslését, a hibajavítást, és segít konzisztens állapotban tartani a követelményeket. Biztonsági szempontból fontos ismerni, hogy az egyes biztonsági követelmények honnan származtathatóak, mi a forrásuk (jogi szabályozás, céges policy, szakértői döntés, stb.), illetve mely más követelményekre van hatásuk.

Ahogyan a szoftver forráskódjánál is nagyon fontos az egyes verziók nyomonkövetése, a követelmények változását is folyamatában kell nyilvántartani, nem elegendő mindig csak a legújabb verziót ismerni. Ennek a nyilvántartásnak két dimenziója lesz, az egyik a követelményeké, a másik a verzióké. Minden egyes követelménynek különféle verziói lehetnek. Egyszerre lehet olyan a követelmények között ami nem változott, és olyan ami már a sokadik verziónál tart, és nyilvánvalóan időben előrehaladva keletkezhetnek, illetve eltűnhetnek követelmények. Definiálhatunk úgyneve-

zett konfigurációkat, amik minden egyes követelmény legfeljebb egy konkrét verzióját tartalmazzák, és ezek között lehetnek kiemelt konfigurációk, úgynevezett baseline-ok (ami alapján például a szoftver 1.0-ás verziója készül).

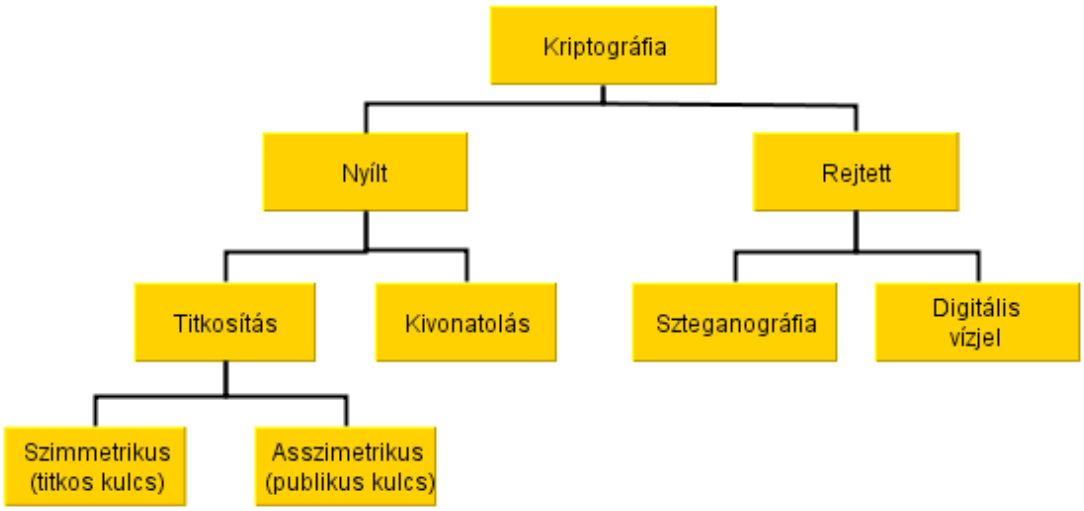
***Fogalmak a 3.6. fejezetben:*** követelmény-menedzsment, Change Control Board, traceability, baseline.

# 3.7 Biztonsági követelmények fajtái

Egy szoftverrel szemben különféle biztonsági szempontok alapján lehet biztonsági követelményeket támasztani. Ezeket az elveket négyféle kategóriába sorolhatjuk: alapvető (core), általános (general), működési (operational) és egyéb (other).

## 3.7.1 Alapvető követelmények

A bizalmassági (**Confidentiality)** követelmények azok, melyek az érzékeny és/vagy titkos adatokhoz való jogosulatlan hozzáférés megakadályozását célozzák meg. Az adat-osztályozás során a rendszer által kezelendő különféle adatot, információt a publikus/nem-publikus címkék valamelyikével is ellátjuk. A nem-publikus adatokat ezek után többféle módszer segítségével védhetjük meg az illetéktelen hozzáféréstől. A maszkolás tulajdonképpen az érzékeny adat felismerhetetlen formában történő megjelenítése (vagy meg sem jelenítése). Nyílt titkosírásról (overt secret writing) akkor beszélünk, amikor az adatot valamilyen módszerrel (pl. kétirányú titkosítással vagy egyirányú hash-eléssel) úgy titkosítjuk, hogy a titkosított verzióból (kétirányú esetben megfelelő dekódolás híján) az eredeti felismerhetetlen legyen.




*14. ábra Kriptográfia típusai* [3]

Rejtett titkosításról (covert secret writing) pedig akkor beszélünk, amikor a titkos üzenetet egy másik, egyébként értelmezhető üzenetbe rejtik (steganography vagy digitális vízjel). A bizalmassági követelményekre az adat szállítása, feldolgozása és tárolása során is figyelnünk kell.

Az integritási (**Integrity**) követelmények azok, amelyek a szoftver megbízhatóságának biztosításával vagy a jogosulatlan módosításának megelőzésével vagy detektálásával foglalkoznak. Egyaránt beszélhetünk a szoftver és az adat sérthetetlenségéről. A megbízhatóság biztosítása alapvetően azt jelenti, hogy a szoftver az elvárt módon működik, míg a rendszer és adatainak jogosulatlan módosításoktól való védelmére a rendszer pontosságaként is hivatkozhatunk. Többféle módszer létezik a sérthetetlenség biztosítására, például az input adatok validációja (többek között

3 Official (ISC)2 Guide to the CSSLP CBK

injection típusú támadások ellen) vagy a paritás bit ellenőrzés alkalmazásai (CRC), de akár a hash is használható változtatás-detektáláshoz.

Az elérhetőségi (**Availability*)*** követelmények a rendszer és adatainak megsemmisülésétől, szolgáltatásainak megtagadásától való védelmet célozzák meg. Ilyen követelményeket a Maximum Tolerable Downtime (MTD), Recovery Time Objective (RTO) segítségével tudunk meghatározni. Előbbi megadja azt a maximális időt, amelynél rövidebb idejű leállásokat a rendszer még komolyabb következmények nélkül elvisel. Utóbbi pedig azt határozza meg, hogy leállás esetén maximum mennyi idő alatt kell újra működőképessé tenni a rendszert. A Recovery Point Objective (RPO) azt határozza meg, hogy a hiba előtti milyen időintervallumba eső adatok elveszése tolerálható. Ezeket az értékeket nemcsak követelményként, de az üzemeltetési szerződésben is fel kell tüntetni.

A hitelesítési (**Authentication**) követelmények azok, amelyek biztosítják, hogy a rendszerrel kapcsolatban álló entitások (felhasználók, egyéb rendszerek) azonosítása (megbízható listában tárolt ismert entitásokkal való összerendelése) rendben megtörténjen. Ez lehet két- vagy többelemű azonosítás. Bármelyikről legyen is szó, biztonsági szempontból erősen ajánlott a létező és elismert technikák használata újak kifejlesztése helyett. Sok létező mechanizmus épül valamilyen azonosító/jelszó páros megadására, ilyen esetekben oda kell figyelni, hogy ezen adatok szállítás közben is biztonságban legyenek. Érdemes külön megemlíteni a biometrikus azonosítást, ahol a biometrikus jellemzők lehetséges változása miatt csak bizonyos pontosságú azonosításról beszélhetünk. Itt meg kell találni az egyensúlyt a tévesen elutasított próbálkozások és a tévesen engedélyezett próbálkozások aránya között.

A jóváhagyási (**Authorization**) követelmények azt határozzák meg, hogy az azonosított entitások a rendszer mely részeihez milyen módon férhetnek hozzá. Ehhez fontos a subject/object (felhasználó/használható programelem) mátrix definiálása, valamint az object-eken végezhető műveletek felsorolása (ezeket szokás CRUD műveleteknek nevezni a Create, Read, Update, Delete szavak alapján). Többféle hozzáférési modell létezik, ezek alapvető típusai a következők:

- *Discretionary Access Control* (DAC), amikor egy-egy programelem tulajdonosa határozza meg, hogy milyen hozzáférést engedélyez az egyes azonosított felhasználók (vagy akár felhasználói csoportok) számára.
- *Ezzel szemben a Non-Discretionary Access Control* (NDAC) esetében nem a tulajdonos, hanem a központi [RE: biztonságpolitika][NOTE: security policy] rendelkezik ezekről a jogokról.
- *Mandatory Access Control* (MAC) esetén a programelemeket felcímkézzük a bennük lévő információ titkossága alapján, és a felhasználókhoz rendelt hozzáférési szintek alapján döntjük el, hogy az elérhető lesz-e a felhasználó számára.
- *Role-Based Access Control* (RBAC) esetén a felhasználók és programelemek közé szerepeket teszünk. Ezek a szerepek határozzák meg, hogy mit tehetünk és mit nem az egyes erőforrásokkal. Egy felhasználónak többféle szerepe lehet.
- *Resource-Based Access Control* során az erőforrást egy harmadik fél éri el közvetlenül, akitől vagy elvárható, hogy maga végezze a felhasználó jogainak ellenőrzését, vagy aki számára a felhasználó (ideiglenesen) átruházza a jogait.

A felelősségi (**Accountability**) követelmények a felhasználói aktivitás naplózására vonatkoznak. Az események loggolása segíthet különféle jogosultsággal kapcsolatos hibák, visszaélések detektálásában, hibák okainak felderítésében. Biztonsági szempontból minden üzleti tranzakcióhoz és adminisztratív funkcióhoz le kell tárolni azt, hogy ki, mikor, milyen elemen milyen műveletet végzett.

Általános biztonsági követelmények közé sorolhatóak azok, amelyek a session-ök biztonságát szavatolják. Egy session arra szolgál, hogy a felhasználó bejelentkezett, azonosított állapotát fenntartsuk előre meghatározott feltételek mellett (adott ideig, kijelentkezésig, program bezárásáig, stb.), ezt egy azonosítás után kiadott egyedi session ID segítségével érjük el. A követelményeknek azt kell szavatolnia, hogy ez a session ID nem jut illetéktelenek kezébe, akik így jogosulatlan hozzáférést kaphatnának a rendszerhez. Fontosak a hibák, kivételek kezelésére vonatkozó követelmények is. Túlságosan részletes hibaüzenetek, vagy nem kezelt kivételek sok mindent felfedhetnek a rendszer belső architektúrájából, működéséből, és ez biztonsági szempontból nem kívánatos. Érdemes tehát megfelelő kivételkezelést és rövid, lényegre törő hibaüzeneteket alkalmazni.

A rendszer beállításainak, paraméterértékeinek tárolására, ellenőrzésére, módosítására vonatkozó követelmények célja, hogy az ilyen jellegű adatok ne kerülhessenek illetéktelen kezekbe, illetve ne lehessen észrevétlenül és jogosultság nélkül megváltoztatni őket.

A szoftver biztonságát a kifejlesztése során alkalmazott és megvalósított biztonsági követelmények nem tudják teljes mértékben garantálni. Sőt, valójában sokkal több biztonsági rést, támadófelületet szolgáltatnak a szoftver működtetésével kapcsolatos hiányosságok. Hiába használ például magas szintű kétkulcsos titkosítással kódolt kommunikációs csatornát a szoftver, ha a privát kulcsot könnyen meg lehet szerezni. Fontosak tehát a rendszer működtetésére vonatkozó követelmények.

A telepítési környezetre vonatkozó követelményeknek olyan kérdéseket kell tisztáznia, mint a hálózati környezet, nyitott portok, terheléselosztás, tárhelyek biztonsága, single sign-on használata, stb.

Archiválási követelmények több okból keletkezhetnek. Az archivált adatokra hasonló biztonsági követelményeket kell megfogalmazni, mint a működési adatokra, de gondolni kell olyan plusz kérdésekre is, mint például a „Pontosan mit?", „Milyen formában?", „Hol?", „Hogyan lesz kereshető?". A szoftverkalózkodás ellen is védekeznünk kell, ha a szoftvert nem házon belülre készítjük. Ide tartoznak például a szoftver licensz-menedzsmentjére, a programkód visszafejtésének, módosításának megnehezítésére vonatkozó követelmények.

Van még pár egyéb követelményfajta, amikre szintén oda kell figyelni.

Az időzítési követelmények nagyon fontosak párhuzamosan futó kódrészletek esetén. Az időzítés rossz kezelésével elveszhetnek események, adatok, a program viselkedése véletlenszerű lehet, esetleg holtpontra juthat. Ezeket a hibákat különféle célzott támadásokkal ki lehet használni, és komoly biztonsági következményeik lehetnek, tehát gondolni kell rájuk a követelmények megfogalmazásánál.

Egy nemzetközileg használt szoftvernél fontos odafigyelni az országonként eltérő jogi környezetre, ezek mindegyikének meg kell felelni. Emellett az eltérő nyelvi sajátosságok is biztonsági problémát jelenthetnek, ha például más-más karakterkódolású szövegeket kell tárolni, erre fel kell készíteni a szoftvert (pl. 20 UTF-8 karakter foglalhat akár 40, vagy több bájtot is a memóriában), de a más-más nyelveken különféleképpen megjelenő adatok egységes formában történő tárolása (kanonizálás, pl. tizedespont és -vessző) is csökkenti a biztonsági kockázatokat.

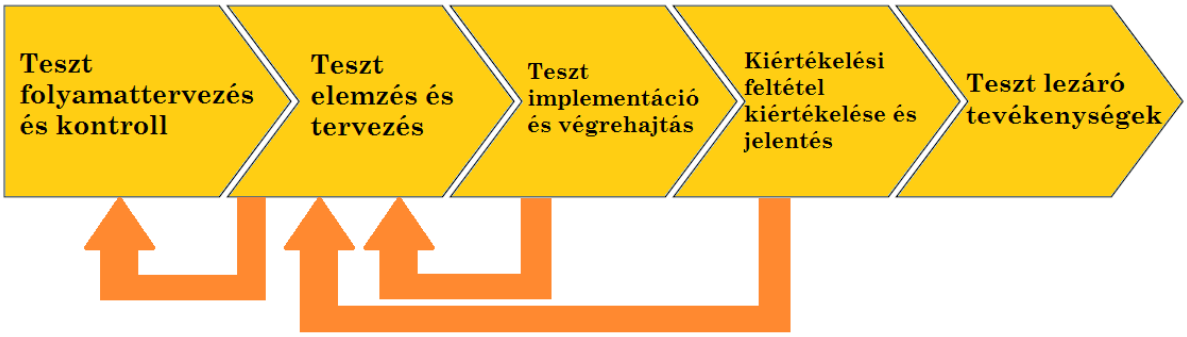
Fontos, hogy a funkcionális követelmények mellett a biztonsági követelményeket akkor is pontosan rögzítsük, ha egy szoftvert nem magunk készítünk, hanem megvásárolunk. Ilyen esetekben fontos a jogi biztonság is, vagyis a követelmények a szerződések részét kell, hogy képezzék.

***Fogalmak a 3.7. fejezetben:*** bizalmassági követelmény, maszkolás, nyílt titkosítás, rejtett titkosítás, integritási követelmény, Maximum Tolerable Downtime (MTD), Recovery Time Objective (RTO), elérhetőségi követelmény, Recovery Point Objective (RPO), hitelesítési követelmény, biometrikus azonosítás, felelősségi követelmény, session, archiválás, időzítési követelmény.

# 4. TESZT MANAGEMENT

## 4.1 Teszt folyamat

A tesztelés sokkal több annál, mint egyszerűen tesztek végrehajtása: a tesztelést elő kell készíteni tervezéssel, a környezet felállításával, valamint utómunkák elvégzése is fontos, melynek során kiértékelésre kerülnek a tesztek és teljesítési kritériumok. Ezen felül számos más kiegészítő tevékenység van, melyek pontosabbá, hatékonyabbá teszik a tesztelést.




*15. ábra Tesztfolyamat lépései*

A teszt folyamata általánosságban az alábbi lépésekre bontható:

### 4.1.1 Teszt folyamattervezés és kontroll

A teszt folyamattervezési fázis során meghatározásra kerül a tesztelés „külalakja": ki, hol, mikor, mit, mivel, hogyan, meddig tesztel. A tesztmenedzser (mások segítségével) meghatározza a tesztelés módját, időpontjait, mérföldköveit, az egyes szerepköröket, tesztelési és teljesítési feltételeket, kockázatokat, stb. A teszt folyamattervezés előtt már meghatározásra kerülnek a tesztelési elvek és a tesztstratégia, amelyek alkalmazására nemcsak az első fázisban kell figyeljünk, hanem a teljes tesztelési folyamat alatt. A teszt kontroll a tervezett lépéseket a valós lépésekkel hangolja össze, kezeli az ezek között fellépő eltéréseket. Ahhoz, hogy a megfelelő korrekciókat végezzük el, az eredmények folyamatos mérése és elemzése szükséges. Mind a tervezés, mind a kontroll a tesztelés befejezéséig folyamatosan tartó tevékenység. Alkalmazásbiztonság szempontjából már ebben a fázisban érdemes számolni a lehetséges projekt és termék kockázatokkal és problémákkal, hogy a tesztelés későbbi szakaszaiban csökkenjen az adatvesztés, azok illetéktelen kezekbe kerülésének, illetve a biztonsági hiányosságokat tartalmazó szoftver release veszélye.

### 4.1.2 Teszt elemzés és tervezés

Ebben a fázisban a tesztesetek részletes meghatározása folyik, amely egyfajta átmenetként szolgál a teszt tervezés és végrehajtás között. A tesztek tervezésének az alapja az ún. teszt bázis, amely magában foglal minden olyan forrást, amely alapján a tesztesetek megtervezhetőek, pl. teszt specifikáció, teszterv, forráskód, tapasztalat, rendelkezésre álló eszközök stb. Bár a specifikációban kellett meghatározni a szoftverrel szemben támasztott követelményeket, biztonsági szempontból a tesztelőknek érdemes visszanyúlni azokhoz a forrásokhoz, amikből a követelmények keletkeztek, és így magukat a követelményeket is validálhatják.

Biztonsági tesztelés szempontjából a teszt bázis része lehet minden olyan forrás, amely a szoftver biztonsági tulajdonságait határozza meg, pl. titkosítási algoritmusok, követelményspecifikáció, valamint más dokumentumok biztonságra vonatkozó része, külső forrásokkal való kapcsolattartásért felelős programrészek, stb. Főképpen üzletkritikus rendszerek esetében a biztonságos működést ellenőrző tesztelés tervezése elengedhetetlen.

### 4.1.3 Teszt implementáció és végrehajtás

Ebben a fázisban a tesztek futtatása, annak előkészületi és utómunkái folynak. Az implementáció része a tesztesetek megalkotása, azok véglegessé tétele, priorizálása, input adatok előállítása, a teszteljárás előállítása és a környezet felállítása. Automatizált tesztelés esetén az implementáció a szkriptírást is magába foglalja. A teszteseteket ezután csoportokba, tesztkészletekbe kell rendezni. Fontos megjegyezni, hogy futtatás előtt a tesztkörnyezetet ellenőrizni kell, hogy készen áll-e a tesztek futtatására. A futtatás folyamán kapott eseményeket/lépéseket, eredményeket naplózni kell, hogy az aktuális és elvárt eredmény könnyen összehasonlítható legyen, valamint hogy a lehetséges okok könnyen feltárhatók legyenek. Az elbukott teszteseteket később érdemes újrafuttatni ellenőrző tesztelés céljából. Biztonsági tesztek végrehajtása, események/incidensek rögzítése biztonsági szempontból kritikus, így ebben a fázisban különösen figyelni kell arra, hogy a felfedezett fenyegetések jelentve legyenek.

### 4.1.4 Kilépési feltétel kiértékelés és jelentés

A tesztelés végrehajtásának befejezése után meg kell vizsgálni, hogy az előre meghatározott kilépési feltétel teljesült-e: ha nem, akkor vagy a feltétel módosítását kell elvégezni, vagy további tesztelést kell végezni. Ezen felül a teljes tesztelési tevékenység ebben a fázisban kerül kiértékelésre, ahol egy összefoglaló jelentés formájában az üzleti vezetés felé továbbíthatóak az elért eredmények.

Az összefoglaló jelentésben nemcsak a funkcionális tesztelés eredményeit kell feltüntetni, hanem a nem formális értékeléseket is, pl. teljesítmény, karbantarthatóság és biztonság szempontjából. A biztonsági tesztelés összefoglalójának, vagy incidens riportnak tartalmaznia kell a talált hiba vagy esemény súlyosságát, előfordulási valószínűségét és egyéb, fontos adatait.

### 4.1.5 Teszt lezáró tevékenységek

A tesztelési folyamat utolsó fázisa, ahol leginkább csak ellenőrzési feladatokat kell elvégezni, így ellenőrizni kell a dokumentációt, az átadandók listáját, a tesztelt alkalmazás verzióiban történt változásokat stb. A lezárási tevékenységek közé tartozik a tesztkörnyezet és infrastruktúra lezárása és archiválása is, az alapkörnyezet átadása a karbantartóknak, valamint a tapasztalatok rögzítése. Biztonsági hibák rögzítésére, elemzésére is oda kell figyelni, hogy azok feltétel nélkül javításra kerüljenek, főleg üzletkritikus rendszerek esetében.

A tesztfolyamatban, habár egymást követő tevékenységekből épül fel, nem kizártak a visszalépések, pl. tesztelemzés és tervezés folyamatában fény derülhet hiányosságokra a folyamattervezésben és kontrollban, amelyek utólagosan kerülnek javításra.

A tesztfolyamat során nemcsak a tesztelt szoftver biztonságának ellenőrzése a cél, hanem magának a tesztfolyamat elemeinek biztonságával is foglalkozni kell. Pl. ha egy bank ügyféladataival kívánunk tesztelést végezni, az éles, eredeti adatbázis adatvédelmi szempontok miatt nem használható, hanem létre kell hozni egy olyan adatbázist, amely nem sért személyes jogokat, megakadályozva, hogy fontos üzleti információk illetéktelen kezekbe kerüljenek. A tesztelésre szánt adatbázisnak olyan típusú, formátumú adatokat kell tartalmaznia, mint az eredeti adatbázisban, természetesen fiktív adatokkal.

A tesztelési folyamat során a résztvevők bizalmas információkkal dolgozhatnak, amelyek csapaton kívül kerülése veszélyeztetheti a projektet, ezért egy etikai kódex kialakítása, szabványok követése és az adatbiztonság fontos szempontot játszanak a tesztelési folyamat kialakításában és végrehajtásában.

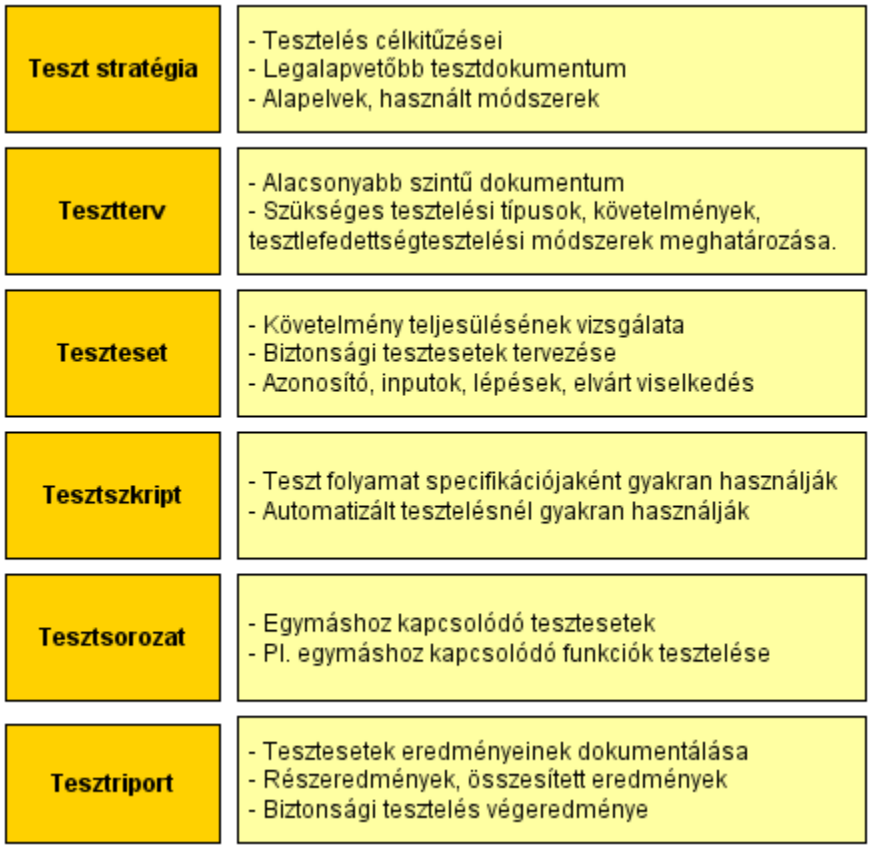
***Fogalmak a 4.1. fejezetben:*** teszttterv, teszt specifikáció, teszteset, kockázat, incidens, teszt bázis, teszt szkript.

## 4.2 Tesztek dokumentálása

A tesztelési folyamatot az elejétől a végéig dokumentálni kell, ennek számos célja van: hogy a létrehozott tervekhez, elvárt viselkedésekhez a kapott eredmények hasonlíthatóak legyenek; hogy sikerüljön betartani a megszabott határidőt az adott erőforrások mellett; hogy a későbbiekben a talált hibák, valamint a szerzett tapasztalat segítsék a fejlesztők és a tesztelők munkáját stb.

A tesztelés során leggyakrabban létrehozott dokumentumok:

- **Teszt stratégia**: a teszt stratégia a tesztelési célkitűzéseket körvonalazza, a legalapvetőbb dokumentum, amely a teszteléssel kapcsolatos fő információkat és alapelveket és használni kívánt módszertant tartalmazza. A teszt stratégia magas szinten meghatározza a teszteléshez szükséges időt, erőforrásokat, környezeti konfiguráció leírását, a tesztek típusait, magasszintű be- és kilépési feltételeket.
- **Teszterv**: a teszterv tulajdonképpen a tesztelési stratégia megvalósításának leírása egy konkrét tesztelési projektben, ahol a megtervezett tevékenységek célja a szoftver vizsgálata. A teszt stratégiánál alacsonyabb szinten taglalja a szükséges tesztelési típusokat, leírva a követelményeket, a konkrét tesztelési módszereket és a tesztlefedettséget. Fő célja a szoftver megbízhatóságának vizsgálata: megfelelt-e a rendszer a követelményeknek, mind funkcionális, mind biztonsági szempontból.
- **Teszteset**: a tesztesetek a követelményeken alapulnak és azt vizsgálják, hogy az adott követelmény teljesült-e, a tényleges viselkedés megegyezik-e az elvárt viselkedéssel. Egy teszteset általában az alábbi részekből áll: egyedi azonosító, hivatkozás a követelményre, előfeltételek, lépések, inputok és elvárt viselkedés. Alkalmazásbiztonság szempontjából nemcsak a funkcionális, de a biztonsági tesztesetek tervezése is igen nagy szerepet játszik.
- **Teszt script**: gyakran használják teszt folyamat specifikációjaként, főként automatizált tesztelésnél.
- **Tesztsorozat (test suite)**: a tesztsorozatok egymáshoz kapcsolódó teszteseteket foglalnak magunkban, pl. egymás után következő funkciók leteszlelése (bejelentkezés, menüpont, almenüpont tartalma).
- **Teszt harness**: egy olyan tesztkörnyezet, amely magában foglalja a tesztelési eszközöket, tesztadatokat, konfigurációkat, teszteseteket és teszt scripteket. Egy tesztelési csonkként is használható, hogy a fejlesztés alatt álló, vagy a felállított tesztkörnyezetben nem elérhető funkcionalitásokat és szolgáltatásokat futtató környezetet szimulálja. Ebből a szempontból a test harness fontos eleme lehet a tesztelés szimulálásának.
- **Teszt riport**: A tesztesetek eredményeinek dokumentálása a tesztriportban történik, amely a mind a tesztelési folyamatokat, mind azok eredményét, egyéb paramétereit rögzíti.




*16. ábra Legfontosabb tesztelési dokumentumok*

A dokumentálás formájára az IEEE 829 tesz ajánlásokat. Az IEEE 829 a szoftver- és rendszer tesztek dokumentációjának sablonjaként szolgál, amely nyolc különböző tesztelési szinten írja le az egyes dokumentumok javasolt sablonját. A definiált dokumentumok:

- **Master Test Plan**: célja egy, a teljes tesztelési folyamatot átfogó terv és teszt menedzsment dokumentáció létrehozása.
- **Level Test Plan**: meghatározza minden egyes tesztelési szinten a célokat, annak megközelítését, a tesztelés körét, a szükséges erőforrásokat és a tesztelési tevékenységek sorrendjét. Ezen felül tartalmazza tesztelendő funkciók halmazát, a részletes tesztelési feladatokat, az egyes tesztelésben szereplők személyes felelősségét és a lehetséges kockázatokat.
- **Level Test Design**: az egyes teszteseteket írja le, azok elvárt eredményeit és a kilépési feltételeket.
- **Level Test Case**: a test design dokumentumban megfogalmazott tesztesetekhez tartozó adatok pontos megfogalmazását tartalmazza.
- **Level Test Procedure**: annak leírása, hogyan futtatjuk majd az egyes teszteket, beleértve az előfeltételeket és az egyes lépéseket.
- **Level Test Log**: a tesztfuttatások lényeges információit tartalmazza időrendi sorrendben, pl. bizonyos tesztesetek mikor lettek futtatva, ki futtatta őket, milyen sorrendben, milyen eredménnyel.
- **Anomaly Report**: minden olyan esemény rögzítésre kerül ebben a dokumentumban, amely további kivizsgálást igényel. Ezek az események lehetnek problémák, incidensek, defektusok, vagy anomáliák. Az anomália riport az incidensek összes adatát tartalmazza, mint pl. az aktuális és az elvárt eredmények, mikor merült fel, és bármilyen más fontos információt, amely hozzásegítheti a tesztelő csapatot a megoldáshoz. Ezen felül még tartalmazhatja a felfedezett incidens lehetséges következményeit.
- **Level Interim Test Status Report**: az egyes tesztszinteken elért időközi eredményeket összegző dokumentum.
- **Level Test Report**: a tervezett tesztttevékenységek eredményeit összegzi, ezzel biztosítva az adott tesztszinthez szükséges eredményeket és ajánlásokat.
- **Master Test Report**: a tesztelési szintek eredményeit összegző dokumentum.

***Fogalmak a 4.2. fejezetben:*** teszt stratégia, tesztterv, teszteset, teszt script, tesztsorozat, IEEE 829.

## 4.3 Teszt típusok

A későbbi fejezetekben részletesen bemutatásra kerülnek az egyes tesztelési típusok, most csak egy kisebb összefoglalót nyújtunk ezek bemutatására. A nagyobb tesztelési típusok a következők:

- Statikus tesztelés
- Specifikáció alapú tesztelés
- Struktúra alapú tesztelés.

Az egyes teszttípusok további altípusokra is bonthatóak.

### 4.3.1 Statikus tesztelés

Statikus tesztelési technikák alatt azon tesztelési technikákat értük, melyeket a program futtatása nélkül tudunk használni, így statikusan teszteljük a forráskódot, a dokumentumokat stb. A statikus technikáknak két fő típusa van: a felülvizsgálat és a statikus analízis. Míg felülvizsgálatot általában manuálisan végzünk dokumentumokon, addig a statikus analízis már leginkább automatizáltan történik, a programkódot vizsgálva.

Mivel statikus tesztelést a tesztelés korai szakaszában visszük véghez, jelentős költségtakarítás érhető el, arról nem is beszélve, hogy a felfedezett és javított hibák biztonsági szempontból is sokat számítanak.

További előnye a módszernek, hogy a kevesebb defektus kisebb fenntartási költséggel jár, valamint a résztvevők közötti kommunikációt fejleszti.

Példa statikusan, felülvizsgálattal felfedezhető hibákra:

- a standardoktól, szabályzatoktól, törvényektől való eltérések, pl. biztonsági követelményekben található anomáliák
- defektusok a követelmények között pl. hiányzó részek, többértelműség
- defektusok a tervezésben, pl. a tesztterv nem felel meg a követelményeknek,

– elégtelen karbantarthatóság, pl. túl nagy vagy komplex kód,
– rossz interfész specifikáció, pl. nem felel meg a tervnek vagy a sémának.

## 4.3.2 Specifikáció alapú tesztelés

A specifikáció alapú tesztelés, más néven feketedoboz tesztelés a tesztelendő rendszer specifikációja vagy valamilyen modellje alapján dolgozik, tehát a forráskód a tesztelők számára ismeretlen. Ebből kifolyólag a működés miként nem ismert, csak azt lehet tudni, hogy az adott program mit csinál. Specifikáció alapú tesztelésnél nemcsak a funkcionalitás, hanem a nem funkcionális tulajdonságok is tesztelendők, mint pl. a használhatóság, teljesítmény és biztonság. A specifikáció alapú technikák közé tartoznak

– az ekvivalencia partíciók,
– a határérték analízis,
– a döntési tábla teszt,
– az állapotátmenet teszt,
– a használati eset teszt,
– a kombinációs tesztelés.

**Kombinációs technikák**

A kombinációs technikákat akkor használjuk, mikor számos paraméter szükséges a szoftver teszteléséhez, ahol az egyes paramétereknek is számos értéke lehet, melyek megnövelik a szükséges kombinációk számát.

A paramétereknek függetlennek és kompatibilisnek kell lennie abban az értelemben, hogy bármely opció bármely együtthatója legyen kombinálható bármely más opció bármely együtthatójával. Az osztályozási fák lehetővé teszik, hogy néhány kombinációt kizárjunk. Ez nem feltételezi azt, hogy a kombinált tényezők nem hatnak egymásra, ami egészen addig nem is probléma, míg ez nem egy elfogadható módon történik.

A kombinációs tesztelés eszközként szolgál a megfelelő kombinációk részhalmazának meghatározására, ezzel biztosítva a kívánt lefedettségi szint elérésének lehetőségét. Számos eszköz létezik a kombinációs tesztelés megvalósítására. Ezen eszközök számára vagy paraméterek és azok értékeinek listája szükséges, vagy azok reprezentációja egy osztályozási fában. A pairwise tesztelés értékek kettős kombinációját alkalmazza a teszteléshez.

## 4.3.3 Struktúra alapú tesztelés

A struktúra alapú tesztelés, más néven fehérdoboz tesztelés a program szerkezetének felderítésére szolgál, a teljes kód ellenőrzéséért felel. A fehérdoboz tesztelés alanyai lehetnek komponensek, hívási függőségek iterációs szinten, valamint egyéb szerkezetek rendszer szinten (pl. üzleti folyamat, menü). A struktúra alapú technikák közé tartozik

– az utasítás teszt és lefedettség,
– a döntés teszt és lefedettség,
– valamin egyéb technikák.

Ezekkel a fehérdoboz teszttervezési technikákkal tulajdonképpen azoknak a teszteknek minőségét mérhetjük, amik a szoftver minőségét mérik.

## 4.3.4 Tapasztalati alapú technikák

A fehér- és feketedoboz tesztelésen kívül léteznek tapasztalat alapú technikák, melyeket leginkább akkor használunk, ha nincs lehetőség szisztematikus tesztelésre. Ez két ok miatt lehetséges:

– a specifikáció alkalmatlan tesztesetek készítéséhez,
– nincs elég idő egy jól strukturált tesztelés elvégzéséhez.

Strukturált tesztelés mellett is hasznos tapasztalati alapú tesztelést végezni.

Mint ahogy az a nevéből is sejthető, a tapasztalati alapú technikák a felhasználók és a tesztelők tudására és tapasztalatára támaszkodnak abból a célból, hogy a rendszer kritikus részeit felderítsék.

### 4.3.5 Fehér- és feketedoboz technikák alkalmazásbiztonság szempontjából

Alkalmazásbiztonság szempontjából a biztonsági tesztelési módszerek között is megkülönböztetünk fekete- és fehérdoboz technikákat.

A fehérdoboz típusú biztonsági tesztelés bemeneteként szerkezeti és tervezési dokumentumok, forráskód, konfigurációs információk és fájlok, használati is visszaélési esetek, teszt adatok, a tesztkörnyezet és biztonsági specifikációk szolgálnak. Mivel a struktúra alapú tesztelésben a tesztelők hozzáférnek a forráskódhoz, könnyen felfedezhetnek kódba ágyazott problémákat is, mint pl. trójai vírust (Trojans), logikai bombákat (logic bombs), megszemélyesítő kódot (impersonation code), kémprogramokat és egyéb kiskapukat a biztonsági rések áttörésére.


*17. ábra Biztonsági fehérdoboz tesztelés folyamata*

A fentebb említett bemeneteket a tesztelők elemzik annak érdekében, hogy megbizonyosodjanak róla, hogy az implementáció követi-e a specifikációt, valamint hogy biztonsági védelmi mechanizmusok vagy sebezhetőségi lehetőségek léteznek-e. A fehérdoboz tesztelés során a rendszer következő elemeit kell vizsgálni: adatfolyam, vezérlési folyam, interfészek, beágyazott kód, bizalmi határok, hibakezelés.

A vizsgálat eredményeit egy riport foglalja magában, amely tartalmazza a talált defektusokat és incidenseket, hiányosságokat és a specifikációtól való eltéréseket, módosítási kérelmeket és ajánlásokat a biztonság javítására.

A biztonsági tesztelés feketedoboz technika már a telepítés előtt elvégezhető, vagy rögtön a telepítés után. Attól függően, hogy mikor szeretnénk elvégezni a tesztelést, a fekete doboz tesztelésnek más-más célja lehet.


*18. ábra A biztonsági feketedoboz tesztelési technika sematikus ábrája*

A telepítés előtt végzett vizsgálatot leginkább arra használjuk, hogy felfedezzük a beüzemelés előtt a biztonsági réseket, így a szoftver feltörésének kockázata minimálisra csökkenthető.

A telepítés utáni feketedoboz tesztelést két okból szokták használni:

1.) Segít megtalálni a biztonsági réseket a telepített csomagban vagy annak futtatási környezetében.
2.) Bizonyítja a szoftver biztonsági funkciók jelenlétét és hatékonyságát.

Minél korábban elkezdjük a feketedoboz tesztelést, annál több karbantartási költséget takaríthatunk meg. A feketedoboz tesztelésre több eszköz áll rendelkezésre, ilyenek pl. a szkennelési technikák és az átható tesztelés.

***Fogalmak a 4.3. fejezetben:*** statikus tesztelés, specifikáció alapú tesztelés, struktúra alapú tesztelés, felülvizsgálat, statikus analízis, biztonsági rés, adatfolyam, vezérlési folyam.

## 4.4 Tesztesetek tervezése

Ahogy már a tesztek dokumentálásánál is leírtuk, a tesztesetek fő része általában az egyedi azonosító, hivatkozás a követelményre, előfeltételek, lépések, inputok és elvárt viselkedés. A tesztesetek tervezéséhez elsősorban ismerni kell a program-specifikációt, mind funkcionális, mind biztonsági szempontból.

Továbbá a tervezésnél figyelembe kell venni, hogy kik fogják majd használni és olvasni őket, nem mindegy, hogy kezdő vagy tapasztalt tesztelők, belső vagy külső munkatársak végzik a tesztelést. A tesztesetekhez tartozó tesztadatok meghatározásánál figyelni kell arra, hogy pontosan a megadott adatokat kell használni, vagy csak példaként szolgálnak.

A tesztek tervezésének három fő lépése van. Először meg kell határozni a tesztelési feltételeket, tehát azt, hogy pontosan mit szeretnénk tesztelni. Második lépésként következik a tesztesetek meghatározása, melynek során megtervezzük a teszteseteket a tesztbázis alapján, valamint ellenőrizzük a tesztelés feltételeit. Befejezésként a tesztelési eljárás specifikációját kell definiálni: hogyan lehetséges a tesztesetek végrehajtása, mik a lehetséges inputok, milyen lépéseket kell végrehajtani és mi az elvárt viselkedés.

Ha az IEEE 829 ajánlást szeretnénk követni, a következőképpen kell felépíteni egy tesztesetet:

1.) Teszteset azonosító: egyedileg generált, számokból álló azonosító, amelynek tartalmaznia kell a tesztelés szintjét, valamint a szoftver verzióját is. Teszteset azonosító létrehozásakor érdemes követni a tesztelendő szoftverben is használt jelöléseket. Az azonosító tehát egy rövid, egyedi névből áll, amelyhez érdemes feljegyezni a teszteset verziószámát és dátumát, a szerzőt és a kapcsolattartási információkat, valamint a revíziók történetét.
2.) Vizsgált elemek: egy tesztesetnek tartalmaznia kell a tesztelendő funkciók és elemek leírását vagy a hivatkozását. Az elem leírását nyerhetjük a
    *a)* követelményspecifikációból,
    *b)* rendszerterv specifikációból,
    *c)* felhasználói kézikönyvből,
    *d)* műveleti leírásokból,
    *e)* telepítési kézikönyvből.
3.) A teszteseteknek tartalmaznia kell a bemeneti specifikációt, amely tartalmaz minden, a teszteset futtatásához szükséges adatot, valamint annak előfeltételeit. Egy teszteset bemeneteként szolgálhatnak konkrét értékek, változók, táblázat, emberi cselekvések, feltételek, fájlok, adatbázisok, kapcsolatok stb. Ezen felül a végrehajtandó lépéseket is definiálni kell, vagyis azon lépések sorozatát, ahonnan eljutunk a bemenettől egészen az elvárt viselkedésig.
4.) Természetesen nemcsak a bemeneti adatokat, hanem az elvárt viselkedést is specifikálni kell: így tudjuk összehasonlítani a tervezett és kapott eredményt, majd eldönteni, hogy a tesztelt elem hibás-e vagy sem. Kimenetként a bemenethez hasonló típusú adatok határozhatók meg. A kilépési feltétel meghatározza, hogy mi a teszteset hivatalos befejezési feltétele. Célja, hogy megakadályozzuk az olyan esetek befejezettnek tekintését, amelyeknek még vannak függőben levő, be nem fejezett részei, vagy hibásan működnek.

Biztonsági teszt felépítése a kulcsszavak felsorolásával kezdődik, amelyek kulcsfontosságúak az adott sebezhetőség tesztelésében. Ezután a megcélzott sérülési lehetőségek típusának felsorolása következik, majd a teszteset lépései. A tesztesetben ezután az elvárt eredmények kerülnek leírásra.

A specifikáció alapú biztonsági tesztelés a következő esetekben használható hatékonyan:

- a munkamenetkezelés megtámadása,
- a hitelelesítési mechanizmusok megtámadása,
- az átirányítási fejlécek manipulálása,
- veszélyes URL-ek engedélyezése,
- rosszindulatú fájlok feltöltése,
- dokumentált jelszavak keresése,
- érzékeny információk bizalmasságának tesztelése,
- beviteli validáció mechanizmusainak támadása.

De ezeken kívül is számos helyzet van, ahol a specifikáció alapú biztonsági tesztelés sikeresen alkalmazható.

***Fogalmak a 4.4. fejezetben:*** követelményspecifikáció, rendszerterv, felhasználói kézikönyv, telepítési kézikönyv, sebezhetőségi teszt.

## 4.5 Teszt folyamat fejlettség

Egy vállalatnál a tesztelés célja az általa piacra dobott termékek és szolgáltatások javítása – viszont nemcsak a termékek, hanem maga a tesztelési folyamat minősége is javítható. Többféle módszer létezik arra, hogy a szoftvertesztelési folyamatok javíthatók legyenek. Ezek a módszerek magának a tesztelési folyamatnak és eredményeknek a javítását célozzák meg, irányelveket biztosítva a fejlődéshez.

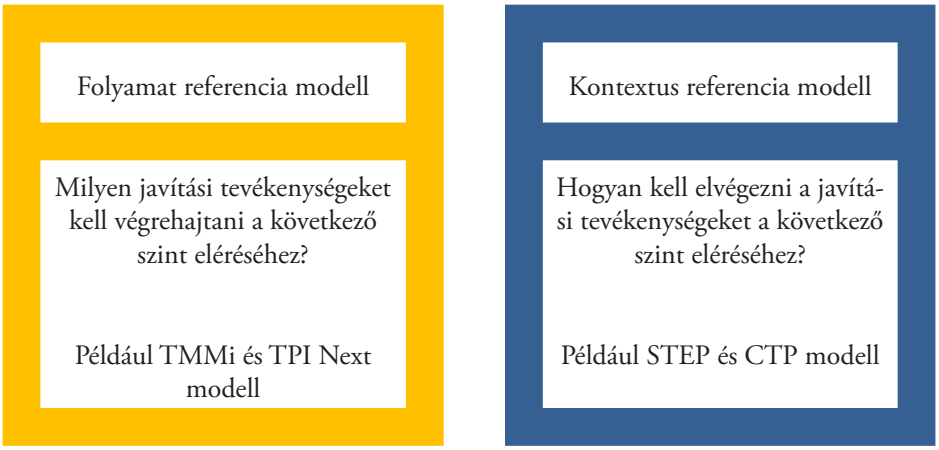
A tesztelési költségek a teljes költségvetés igen nagy részét képezik, mégis csak kevesen figyelnek a tesztfolyamat fejlődésére annak ellenére, hogy számos modell létezik már a tesztelés javítására, ilyen pl. a CMMI, Test Maturity Model integration (TMMi), Systematic Test and Evaluation Process (STEP), Critical Testin Processes (CTP) és TPI Next. Ezeket a modelleket megfelelően használva metrikákat nyerhetünk, melyekkel az egyes szervezetek összehasonlíthatóak.

A folyamatok fejlesztése legalább olyan fontos a szoftverfejlesztésben, mint a tesztelési folyamat. Egymás hibáiból tanulva, a tapasztalatokra alapozva a fejlesztési és tesztelési szoftverek javíthatóak. A Deming fejlesztési módszere a következő lépésekből áll: tervezz, cselekedj, ellenőrizz, alakíts (Plan, Do, Check, Act). A technikát már évtizedek óta használják a szoftvertesztelők folyamatok javítására.

A folyamat modellek egyfajta kezdőlépésként szolgának a tesztelési folyamat javításának elkezdésében, összemérve a szervezet folyamatbeli képességeit a modellel.

Értékelési modellek használata egy bevett gyakorlat, amely egy szabványos megközelítést nyújt tesztfolyamatok fejlesztésére.

A fejlesztési modelleknek két fajtája van, ezek a folyamat referencia modellek és a tartalmi referencia modellek.



### 4.5.1 Folyamat referencia modellek

Folyamat referencia modellek, melyek fejlettségi mérést biztosítanak az értékelési folyamat részeként abból a célból, hogy összehasonlítsák a szervezet képességeit a modellel, hogy értékeljék a szervezetet egy adott keretrendszeren belül, valamint hogy egy ütemtervet biztosítsanak a javítási folyamat számára.

Folyamat referencia modell pl. a TMMi, teljes nevén Testing Maturity Modell integration, amely célja a CMMi kiterjesztése. A modell szintekből áll, ahol minden egyes szint pontosan definiált folyamat területeket tartalmaz. Minden egyes fejlettségi szintnek legalább 85%-osan befejezettnek kell lennie, amelyhez speciális és általános célok elérése egyaránt szükséges ahhoz, hogy a következő szintre léphessünk.

### 4.5.2 Tartalmi referencia modellek

Tartalmi referencia modellek, amelyek egy szervezet lehetőségeinek javítására nyújtanak üzleti alapon végzett értékeléseket, beleértve azokat az eseteket, ahol a benchmarki és ipari átlagok kerülnek összehasonlításra objektív mérésekkel. Tartalmi referencia modell például a CTP, teljes nevén a Critical Testing Processes modell, amely előfeltétele az, hogy az aktuális tesztelési folyamat kritikus legyen. Ha ezeket a kritikus folyamatokat gondosan visszük véghez, az támogatni fogja a tesztelő csapat munkáját. Ezzel szemben, ha nem elég precíz a végrehajtás, még a legképzettebb tesztelők sem lesznek képesek jó munkát végezni.

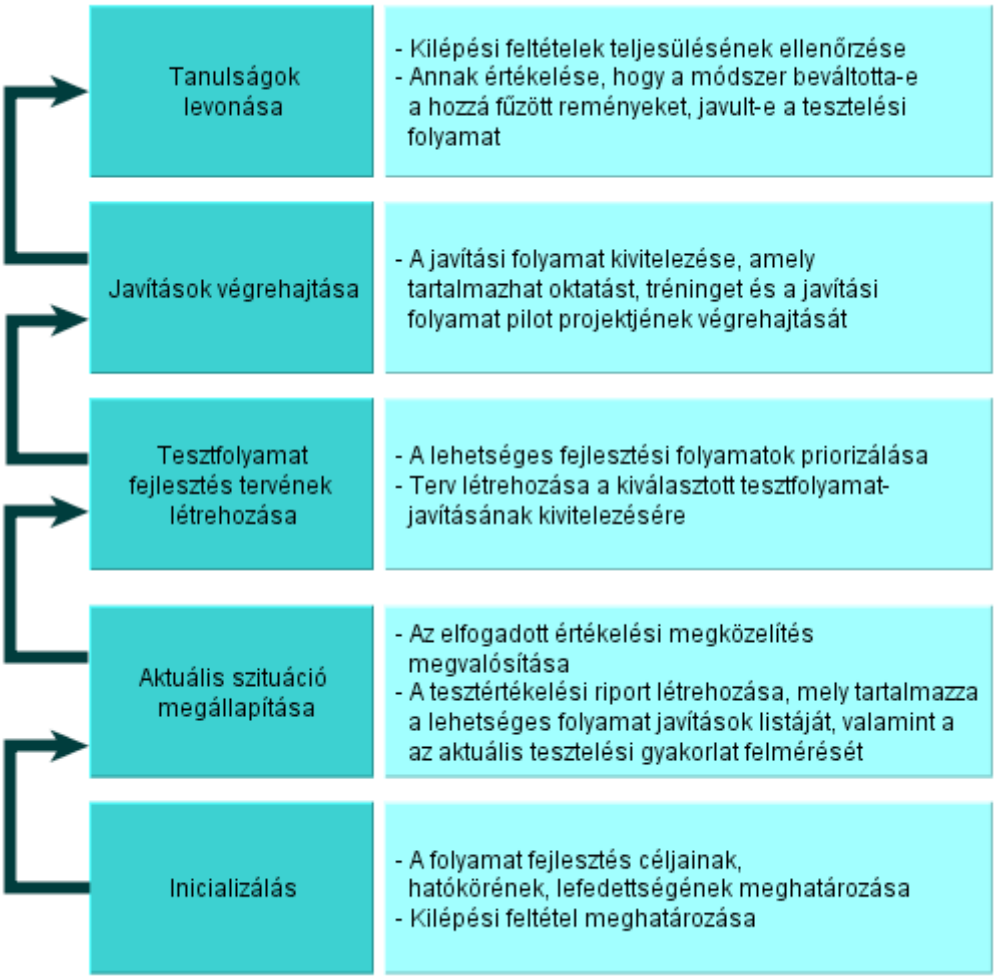
A CTP modell egy kontextus-érzékeny megközelítés, amely segítségével a modell formálható a következők szerint:

1.) speciális kihívások azonosítása, pl. szoftverek feltörése, ellenséges támadások elleni védekezés,
2.) a megfelelő folyamatok elfogadása, elismerése, pl. autentikációs folyamatok fontosságának elismerése,
3.) a fejlesztési folyamat lépéseinek, fontosságának és sorrendjének szabad meghatározása, pl. biztonsági tesztelés előbbre hozása, vagy alaposabb elvégzése, főleg üzletkritikus szoftverek esetében. A CTP tizenkét kritikus tesztelési folyamatot azonosít, melyek közül a szoftverbiztonságot a minőségi kockázatok elemzése képviseli.

### 4.5.3 IDEAL modell

Amint egy cég, vállalat elhatározta, hogy javítani fog a tesztfolyamaton, a következő (IDEAL) lépéseket kell megtennie:

1.) **Fejlesztési folyamat inicializálása**: mielőtt hozzálátnánk a tesztelési folyamatok elvégzéséhez, meg kell határozni a célokat, hatókört és lefedettséget, majd együtt kell dönteni az érintettekkel. A modell fajtája is ebben a fázisban kerül kiválasztásra. Modellt nemcsak a meglévők között lehet választani, az egyes vállalatok saját módszert is kidolgozhatnak a tesztelési folyamat javítására.
2.) **Jelenlegi szituáció diagnosztizálása**: a kiválasztott értékelési módszert a résztvevők elfogadják, majd értékelési riportot készítenek, amely tartalmazza a jelenlegi tesztelési gyakorlatot és a lehetséges javítások listáját.
3.) **A tesztelési folyamat fejlesztési tervének létrehozása**: a lehetséges javítási módszerek listáját priorizálni kell, amely a befektetések megtérülése, kockázatok, szervezti stratégiához való igazodás alapján kerül kiszámításra. Amint a priorizálás megtörtént, elkészül a javítások kivitelezéséhez szükséges terv is.
4.) **Javítási tevékenységek végrehajtása**: ebben a fázisban kerülnek implementálásra a teszt folyamat javítási tervének lépései. A lépések tartalmazhatják a tréningeket, a pilot folyamatokat és a teljes fejlesztést.
5.) **Fejlesztési program összefoglalása, tanulságok levonása**: miután a folyamat javítási lépések implementálása megtörtént, ellenőrizni kell, hogy mely előnyöket, javulásokat értük el. Továbbá fontos, hogy ellenőrizzük a siker kritériumat az egyes folyamatfejlesztési tevékenységeknél.




*19. ábra Az IDEAL modell*

***Fogalmak a 4.5. fejezetben:*** kockázat, CMMI, TMMi, STEP, CTP, TPI Next, IDEAL.

# 5. STATIKUS TESZTELÉS

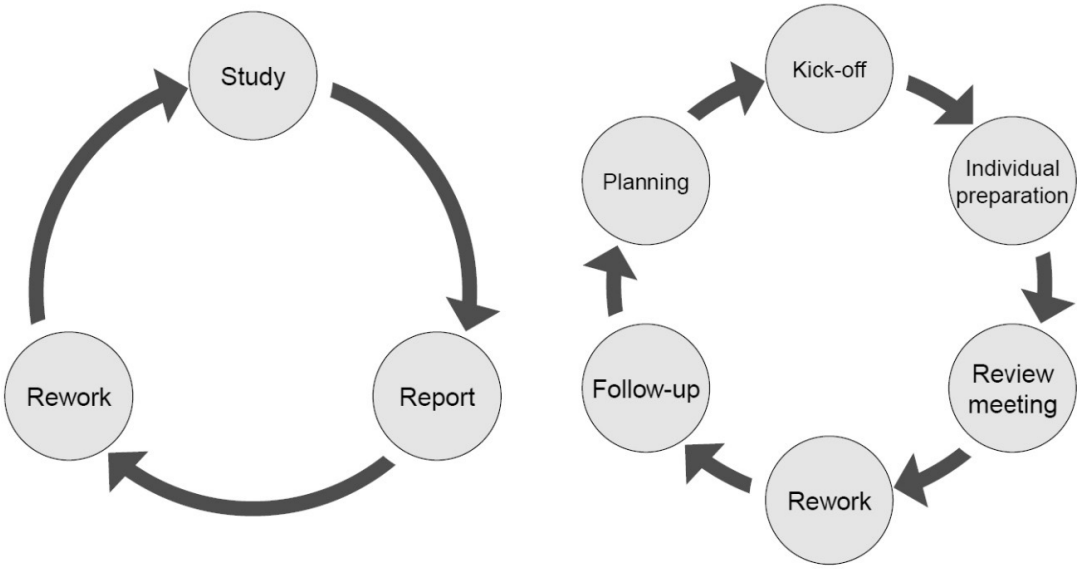
## 5.1 Reviewk

Amint azt az előző fejezetben is említettük, a statikus tesztelés a program futtatása nélkül vizsgálja a forráskódot, a dokumentációt, valamint bármely írott forrást. A statikus tesztelés folyhat manuálisan (felülvizsgálat) és automatizáltan (statikus analízis).

A felülvizsgálat munkatermékek (általában dokumentumok, forráskód) manuális átnézése, vizsgálata hibák felfedezésének céljából. Felülvizsgálatot leginkább követelmény specifikáción, rendszerterven, teszterven, tesztesen, kódon stb. végeznek. A tesztelési folyamatban ez az első vizsgálat, hiszen a tervek már a forráskód előtt elkészülnek. A felülvizsgálattal megtalált hibák általában a következők lehetnek:

- a szabványoktól, törvényektől való eltérések, ilyen pl. a biztonsági követelményekben talált hiba,
- hibák a követelmények között, pl. hiányzó részek, nem egyértelmű megfogalmazás, jogosultságok nem pontos deklarálása, autentikáció hiánya bejelentkezést igénylő funkcióknál, nem megfelelő titkosítás választása,
- hibák a tervezésben, pl. a terv nem felel meg a követelményeknek, vagy nem áll rendelkezésre a tervezett idő, erőforrás. Biztonsági szempontból ilyen lehet egy titkosításhoz értő, biztonsági tesztelésben tapasztalt tesztelő hiánya, a biztonsági követelményeknek nem megfelelő tervezés stb.,
- elégtelen karbantarthatóság, pl. robosztus kód, nem használt kódrészek, vagy olyan kódrészek publikusak, amely láthatóság biztonsági szempontból veszélyt jelent.

A felülvizsgálat többféle is lehet annak függvényében, hogy mi a célja, illetve mennyire formálisan végezzük azt el.




*20. ábra Felülvizsgálatok típusai*

A review típusának kiválasztásában szerepet játszik a fejlesztés előrehaladottsága, a szabályozási követelmények és az auditálás szükségessége. A legalapvetőbb felülvizsgálati folyamat három lépésből áll, melynek során a vizsgáló átnézi a dokumentumokat, majd értesíti a szerzőt a felfedezett problémákról és hiányosságokról. Utolsó lépésben a szerző válaszol a megjegyzésekre, valamint kijavítja a talált hibákat.

Azokban a helyzetekben, mikor formális felülvizsgálatra van szükség, már részletesebb lépések megtételére van szükség, amelyet általában hat lépcsőben végeznek el:

1.) **Tervezés**: ebben a fázisban kerülnek kiválasztásra a felülvizsgálat vezetője és a résztvevők, a be- és kilépési feltételek, valamint az átnézendő dokumentumok listája.
2.) **Indítás**: az indítás során a résztvevők megkapják a felülvizsgálandó dokumentumokat, ellenőrzik a belépési feltételek, valamint kihirdetik a review céljait.
3.) **Egyéni felkészülés**: ebben a szakaszban mindenki egyénileg áttanulmányozza a kapott dokumentumot, majd ezekből jegyzeteket készít.

4.) **Felülvizsgálati megbeszélés**: az egyéni felkészülés során talált hibákat vitatják meg a résztvevők, valamint azok megoldására javasolnak különböző javítási lehetőségeket. Az alkalmazott megközelítést mindig a felülvizsgálat kezdetekor állapítják meg a résztvevő, a vezető jóváhagyásával.
5.) Átdolgozás: az átdolgozás során a defektusok javításra kerülnek a megbeszélések alapján. Minden dokumentumot annak szerzője javít. Fontos, hogy javítás után a defektusok és a dokumentum státusza frissítésre kerüljön.
6.) **Utómunkálatok**: az utolsó fázisban a javításokat és a kilépési feltételt kell ellenőrizni, valamint összeállítani a megfelelő metrikákat (pl. talált hibák száma, reviewre eltöltött idő).

A felülvizsgálatnak több típusa van, a megfelelő kiválasztásában a dokumentum típusa, rendelkezésre álló erőforrások játszanak szerepet. A formalitás növekedésével az alábbi típusokról beszélhetünk:

– **Informális**: nem formális eljárás, dokumentációra nincs szükség. Speciális szaktudásra, ismeretekre sincs követelmény. Viszonylag gyors eljárás, ahol a legfőbb cél a defektusok megtalálása. Adatbiztonság szempontjából nem a legjobb megoldás, ugyanis az informális felülvizsgálás nem szisztematikus, valamint dokumentáció hiányában a pl. különböző verziók követése nem megoldott.
– Átvizsgálás: a megbeszélést a szerző vezeti, a résztvevők pedig a fejlesztőcsoportból kerülnek ki. Szabadon választható, hogy a felülvizsgálat formális vagy informális legyen, legyenek megbeszélés előtti felkészülések, válasszanak-e írnokot. Hasonlóan az informális felülvizsgálathoz itt is a fő cél defektusok feltárása, viszont az átvizsgálásnál ez tipikusan egy forgatókönyv végigjátszásával történik.
– **Technikai felülvizsgálat**: szintén lehet formális vagy informális, viszont a technikai felülvizsgálatot már dokumentálni kell. A szereplők technikai szakértők és fejlesztők, akik egy előre megválasztott defektuskereső eljárást használva derítik fel a lehetséges hibákat. A résztvevők felkészülten, előzetes jelentéssel indítják a felülvizsgálatot. A technikai felülvizsgálat céljai közé tartozik a talált problémákkal kapcsolatos döntések meghozatala, a lehetséges alternatívák kiértékelése, a specifikációnak, standardoknak (és biztonsági előírásoknak) való megfelelés ellenőrzése.
– **Inspekció**: más néven teljes átvizsgálás, melyet képzett moderátor vezet egy olyan résztvevői körben, ahol mindenkinek jól meghatározott szerepe van. Az eljárás maximálisan formális, ellenőrzőlistákkal, be- és kilépési feltételekkel pontosan meghatározva. Az egyéni felkészülés igen fontos, csak precíz felkészüléssel lehet részletekbe menő jelentést készíteni, a javítást segítő metrikákkal. Inspekciónál a cél már nemcsak a defektusok megtalálása, hanem magának az átvizsgálási folyamatnak a javítása.

Az adatbiztonság tesztelése szempontjából a fent említett felülvizsgálati módszerek közül a technikai felülvizsgálat vagy az inspekció a legjobb választás. A felülvizsgálat során fény derülhet sérülékeny és kihasználható kódolási sablonokra, mint pl. az bemenet validációjának, a kimenet kódolásának hiánya, lekérdezések dinamikus építése, közvetlen paraméter manipuláló referenciák használata, nem biztonságos API-k használata stb. Ráadásul a felülvizsgálat akkor is hasznos, mikor rosszindulatú kódok hiányát és a beültetett fenyegetéselhárítást kell vizsgálni. A biztonsági felülvizsgálásoknak ki kell terjedni a szállított szoftver összes aspektusára, beleértve a kódot, komponenseket és a konfigurációkat.

Ezek a felülvizsgálatok elvégezhetők automatikusan vagy kézzel is. Az automatikus kódfelülvizsgálat azért népszerű, mert sokkal skálázhatóbbak és kiterjedtebb kód lefedettséget képesek elérni, mint a manuális felülvizsgálatok.

A biztonsági-technikai felülvizsgálat stratégiára lássunk néhány példát:

– Végrehajtani a felülvizsgálatot egy módosított kódon mielőtt azt jóváhagyjuk,
– Végrehajtani a felülvizsgálatot egy bejárt útvonalon,
– Összefogni a szállító fejlesztő csapatával. Mikor egy csapat együtt dolgozik a felülvizsgálat végrehajtása során, nagyobb eséllyel találnak kevésbé nyilvánvaló sematikus logikai tényezőket, a nyilvánvaló szintaktikai kód gyengeségekkel és veszélyekkel szemben.
– Dokumentáljuk a felfedezett sebezhető kódrészeket és rosszindulatú fenyegetéseket egy adatbázisban, hogy azok nyomon követhetőek és kereshetőek legyenek.

*Forrás: Official (ISC)2 Guide to the CSSLP CBK, Second Edition.pdf, 611-612. oldal*

***Fogalmak a 5.1. fejezetben:*** statikus tesztelés, felülvizsgálat, statikus analízis, követelmény specifikáció, rendszerterv, tesztterv, teszteset, karbantarthatóság, informális felülvizsgálat, átvizsgálás, technikai felülvizsgálat, inspekció.

## 5.2 Statikus elemző eszközök

A statikus tesztelésnek két fő típusa van: a review-k és a statikus analízis. Az előző fejezetben megismerkedtünk a felülvizsgálattal, melyeket leginkább manuálisan végezünk dokumentumokon. A statikus analízis viszont már automatizáltan történhet, célpontjában a forráskóddal vagy modellel. A statikus analízist éppen ezért általában elemzőeszközök végzik, melyek fő használói a fejlesztők. A statikus analízist gyakran használják a komponens tesztelés részeként, vagy a konfigurációmenedzsment eszközbe való küldés előtt.

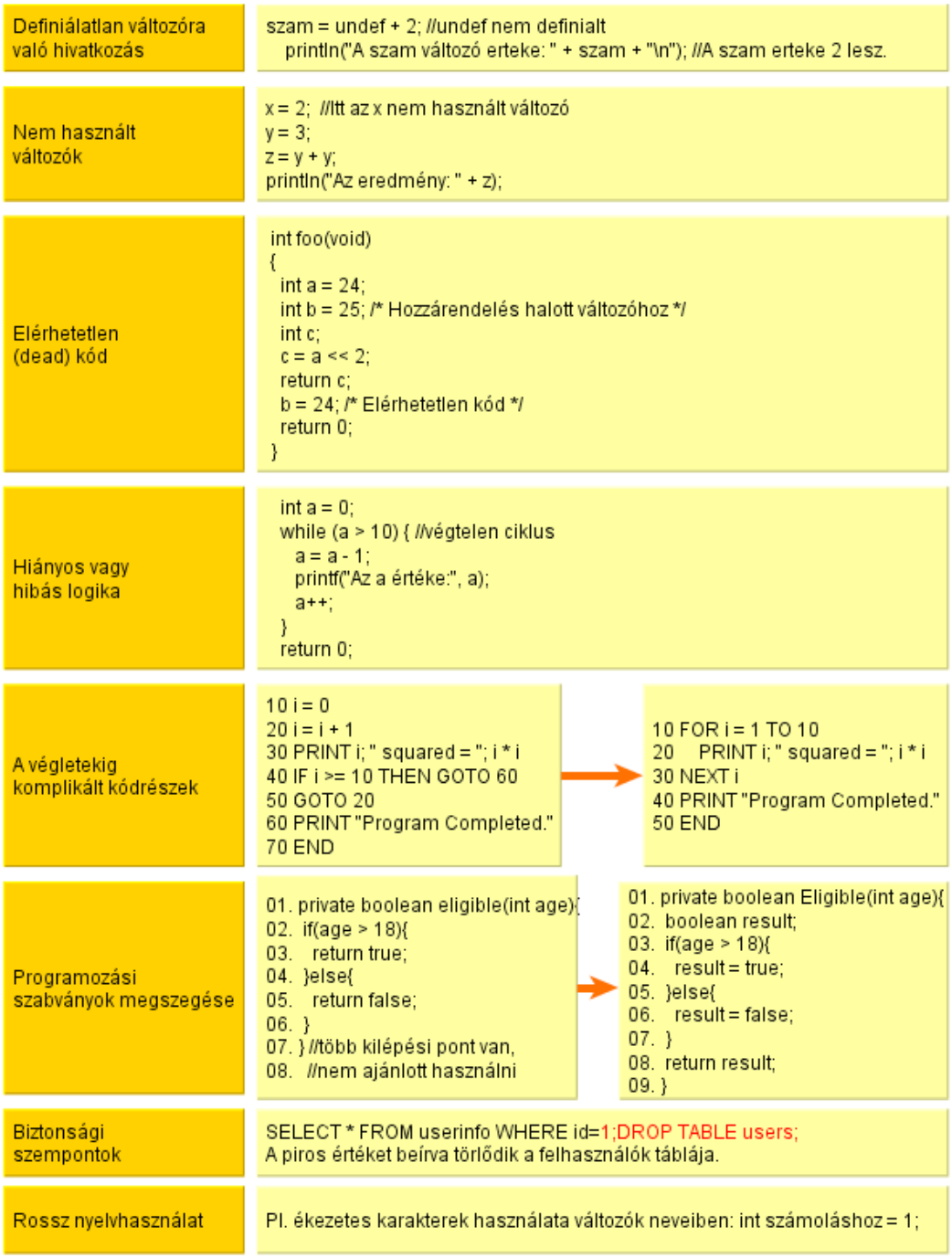
A statikus analízissel leginkább a következő típusú hibák fedezhetők fel:

- definiálatlan változóra való hivatkozás (warning: ‚X' is used uninitialized in this function),
- modulok közötti kapcsolatok hibái, pl. elavult interfész használata,
- nem használt változók (warning: unused variable ‚X'),
- elérhetetlen (dead) kód (warning: comparison is always false due to limited range of data type),
- hiányos vagy hibás logika, pl. végtelen ciklus,
- a végletekig komplikált kódrészek,
- programozási szabványok megszegése, pl. javadoc kommentek hiánya,
- biztonsági szempontok, pl. SQL injection,
- rossz nyelvhasználat (warning: suggest parentheses around assignment used as truth value).

A statikus elemző eszközök kódolási stílust és irányelveket ellenőrizhetnek, amelyek alapján az elnevezési konvenciók, kommentelés, és kódmodularizáció helyessége vizsgálható. Fontos megjegyezni, hogy ezek az elemző eszközök nem a konkrét hibákat képesek felfedezni, hanem úgynevezett figyelmeztetéseket, melyek a kód szintaktikai hibáira hívják fel a figyelmet.

Alapos tervezéssel egy sokkal átláthatóbb, karbantarthatóbb kódot kapunk, amelynek kialakításához a statikus elemző eszközök a következőképpen járulnak hozzá:

- Ismételt kódrészek kutatása. Ezen részek lehetséges, hogy a modulokra való szervezés fő szereplői lesznek.
- Olyan méréseket végeznek, olyan metrikákat generálnak, melyek a kódmodularizáció értékes alapjai lehetnek. Ilyen metrikák pl. a párosításhoz és kohézióhoz tartozó mérések eredményei. Az olyan rendszerek, amelyek nem túl könnyen karbantarthatóak, sokkal nagyobb valószínűséggel rendelkeznek alacsony szintű párosítási mutatóval és magas kohéziós szinttel. Az alacsony szintű párosítási mutató azt jelenti, hogy csak kis mértékben használják egymást a modulok, míg a magas kohéziós szint a csak egy-egy feladatot ellátó, teljesen független modulok magas számára utal.
- Megmutatják, hogy egy objektum-orientált kódban hol túl szigorú vagy túl enyhe a láthatósága a szülő osztályoknak.
- Kiemelik a kódban azokat a komplex részeket, amelyek a rossz karbantarthatóságot okozhatják, vagy nagyobb aránnyal tartalmazhatnak hibákat. A ciklomatikus komplexitás elfogadható szintjét előre meghatározzák (vagy meghatározza a felhasználó), hogy kiszűrhetőek legyenek a komplex kódrészek, melyeket modularizálni érdemes.




*21. ábra Tipikus hibák*

A statikus elemző eszközök között olyanok is kerültek a piacra (mint pl. az IBM Security AppScan Source), melyek kifejezetten a szoftverbiztonsági rések felkutatását végzik, már a fejlesztés korai szakaszában. Mit kell az ilyen biztonsági statikus elemző eszközöknek tudniuk?

- azonosítani a forráskódban levő biztonsági réseket és hibákat,
- a forráskód biztonsági elemzését képes legyen automatikusan is véghezvinni,
- a biztonsági irányelveket fontosság szerint rendezni, a rögzített biztonsági hibákat priorizálni,
- rövid idő alatt minél nagyobb kódot legyen képes átvizsgálni.

***Fogalmak a 5.2. fejezetben:*** statikus tesztelés, felülvizsgálat, statikus analízis, komponens tesztelés, konfigurációmenedzsment, SQL injection, kódmodularizáció, párosítási mutató, kohéziós szint.

## 5.3 Biztonsági elemző eszközök

Előző fejezetben olyan eszközökről beszéltünk, amelyek a statikus tesztelést, ezen belül is a statikus analízist segítették. Most olyan eszközöket fogunk bemutatni, melyek a biztonsági tesztelést támogatják.

A biztonsági tesztelés két fő aspektusban tér el tesztelés többi fajtájától:

- A standard tesztelési technikák a tesztadatok kiválasztásánál valószínűleg fontos biztonsági tényezőket hagynak figyelmen kívül.
- A biztonsági rések és meghibásodások következményei merőben eltérnek más tesztelés által talált defektusok következményeitől.

A biztonsági teszteléshez használható eszközöknek több típusa van, amelyek külön célokra használhatók fel:

- Felderítő, információgyűjtő teszteszközök
- Sebezhetőség vizsgálók
- Mintavételező eszközök
- Protokoll elemzők
- Jelszó feltörő eszközök
- Webbiztonsági eszközök
- Vezeték nélküli hálózat biztonságát tesztelő eszközök
- Fordított fejlesztő eszközök
- Forráskódelemzők
- Biztonsági részek kiaknázására tervezett eszközök
- Biztonság-orientált operációs rendszerek
- Adatvédelem tesztelését segítő eszközök

Felderítő eszköz pl. a Ping, amely az Internet Control Message Protocol (ICMP) segítségével megvizsgálható, hogy egy adott hoszt elérhető az Internetek keresztül avagy sem. Továbbá használható még a lokális hoszttól a célgépig történő csomagküldés idejének mérésére.

Másik felderítéshez használt biztonsági tesztelő eszköz a Traceroute, amely megbecsüli a célszerverhez tartó utat ICMP Echo Request üzenet küldésével, így növelve a Time to Live (TTL) mező értékét. Továbbá ki tudja számítani, mely hosztok továbbítják a csomagokat, így azok megcímezhetőek.

A sebezhetőség vizsgálók közé tartozok a Network Mapper (Nmap), amely egy igen népszerű, ingyenes és nyílt forráskódú hálózati felfedező és biztonsági auditáló eszköz. Nyers IP csomagokat használ, hogy meghatározza a hálózaton elérhető szervereket, valamint operációs rendszerek mintavételezhetőek vele, a szerveren futó szolgáltatások nevét és verziószámát is képes meghatározni, valamint a csomagszűrők és tűzfalak felismerésére is képes.

A sebezhetőséget vizsgálja szintén a Nessus nevű program, amely kliens-szerver architektúrában került implementálásra. Grafikus felhasználói felülettel rendelkezik, és több mint 20 000 pluginnal, amelyek különböző sebezhetőségi lehetőségek feltárását támogatják. Mind Windows, mint UNIX operációs rendszerre készült verziója. Legjellemzőbb funkciói a távoli és helyi (hitelesített) biztonsági tesztelés, valamint a saját programozási nyelv, a NASL, amellyel a tesztelők elkészíthetik saját pluginjüket.

A mintavételező eszközök közül az egyik legismertebb alkalmazás a P0f v2, amely egy leleményes, passzív, operációsrendszert mintavételező eszköz, amely a célhoszton levő operációs rendszert képes felismerni az elkapott csomagok segítségével. Nem generál plusz forgalmat. További funkciója, hogy tűzfalak, Network Address Translation (NAT) vagy terheléselosztók jelenlétét is érzékeli.

A protokoll elemzők közül a legismertebb a Wireshark nevű, nyílt forráskódú program, amely mind vezetékes, mind vezeték nélküli hálózatok elemzésére alkalmas. Részletes információkat képes leszedni a továbbított csomagokról. Képes meghatározni a generált forgalmat a hálózat vagy az azon futó alkalmazások által használt protokollok segítségével, valamint felfedezni a biztonsági problémákat.

Szintén protokoll elemző eszköz a Tcpdump és a WinDump. A Tcpdump a hálózaton átküldött TCP/IP csomagokat kapja el és dumpolja. Igen sok operációs rendszerre készült verziója, ebből a Windowsra készült alkalmazás a WinDump nevet kapta.

Jelszófeltörő alkalmazások közül az egyik legelterjedtebb a Cain & Abel nevű program. Bár a Cain & Abel egy rendívül erős és ismert jelszó feltörő eszköz, amely a jelszavak felderítésére szótárat, teljes végigpróbálást és kriptográfiai elemzéseket használ, még számos más funkcióval is rendelkezik. Képes

VoIP beszélgetéseket rögzíteni, vezeték nélküli hálózati kulcsokat felderíteni, kódolt jelszavakat megfejteni, jelszódobozokat és gyorsítótárban tárolt jelszavakat felfedni és útvonal-megállapító protokollokat elemezni.

A weboldalak biztonságának tesztelésére számos eszközt fejlesztettek, nézzük ezek közül a Nikto2 nevű alkalmazást. A program egy nyílt forráskódú webszerver vizsgáló, amely átfogó tesztelést végez a webszervereken veszélyes fájlok, elavult webszerver verziók és potenciális biztonsági rések után kutatva. Továbbá beazonosíthatók a Nikto2

segítségével az installált webszerverek és az azokon futó alkalmazások amelett, hogy a szerver konfigurációs elemek, mint pl. a többszörös index fájlok és HTTP szerver beállítások is leellenőrizhetők.

A fent felsorolt típusokon kívül még számos más biztonsági aspektust vizsgáló eszköz van, ilyenek a vezeték nélküli hálózat biztonságát tesztelő eszközök, a fordított fejlesztő eszközök, a forráskódelemzők, a biztonsági részek kiaknázására tervezett eszközök, a biztonság-orientált operációs rendszerek és az adatvédelem tesztelését segítő eszközök.

***Fogalmak a 5.3. fejezetben:*** biztonsági tesztelés, biztonsági rés, adatvédelem, kriptográfia.

## 5.4 Folyamatos monitorozás

A monitorozás a rendszer viselkedésének folyamatos figyelését, pillanatfelvételek készítését jelenti. Összesen öt dimenziót említhetünk, amelyeket a tesztfolyamat során monitorozni tudunk:

- termékkockázat,
- defektusok,
- tesztek,
- lefedettség,
- megbízhatóság.

A termékkockázatok, defektusok, tesztek és a lefedettség gyakran mérhető és rögzíthető a fejlesztési és tesztelési folyamat során. Habár a megbízhatóság leginkább felmérések segítségével mérhető, csak szubjektív képet adhat annak eredményéről. A folyamatos monitorozás azért is fontos tényező a tesztelésben, mert információkat nyújt a tesztelésben résztvevők számára a rendszer állapotáról. A begyűjtött adatok pontosságára oda kell figyelni, ugyanis a pontatlan adatok pontatlan, nem megfelelő következtetésekhez vezethetnek. A helyzetet súlyosbíthatja, hogy a pontatlan következtetéses hibás menedzseri döntéseket eredményeznek, ezzel károsítva a teljes tesztelő csapat munkájának hitelességét.

Mint ahogy az azonosított kockázati elemeknél is szokás, a teszteseteket is érdemes követelményekhez kötni. Ez a monitorozás szempontjából is fontos. Vegyünk egy példát:

Ha a tesztelemző tudja, hogy „A” teszteset „A” követelményhez tartozik, és ez az egyetlen „A” követelményhez köthető teszteset, „A” teszteset sikeres futtatásával az „A” követelmény teljesítettnek, megvalósítottnak tekinthető. A legtöbb esetben egy követelmény lefedéséhez több teszteset szükséges, de erőforrások szűkében szükség lehet a tesztesetek halmazának szűkítésére. Például, ha 20 teszteset futtatására van szükség egy követelmény lefedésére, de csak 10 teszteset lett létrehozva és futtatva, a lefedettségi mutató 100%-ot jelezhet, valójában a követelmény lefedettsége csak 50%.

Ilyen eszköz pl. a SonarQube, amely egy nyílt forráskódú platform a kód minőségének folyamatos vizsgálatára. A SonarQube a kódminőség 7 dimenzióját fedi le:

- kommentek,
- kódolási szabályok,
- potenciális bugok
- komplexitás,
- Unit tesztelés,
- duplikációk,
- architektúra és tervezés.

A program pluginok segítségével több mint 20 programozási nyelv elemzésére képes, valamint számos plugin szolgál további szabályok, metrikák használatának bevezetésére. A CodeAnalyzer plugin egy ingyenes kódelemzési megoldás SonarQube felhasználóknak. A plugin jelenlegi verziója a beépített Java kódelemző motorját terjeszti ki több szofisztikált és precíz metrikával, méréssel.

Hogy pontosan milyen új funkciókat biztosít a CodeAnalyzer plugin, a következő lista sorolja fel:

- Egy részletes és precíz java forráskódelemző motor.
- Osztály és metódus szintű elemzések és drill-down nézetek fájl alapú megközelítésben.
- Kifinomultabb kettes típusú klónkereső motor.
- További precíz forráskód metrikák mind általánosságban, mind klónokra vonatkozóan.
- Újrapriorizált és gondosan válogatott PMD szabálysértés-gyűjtemény.
- Kódolási szabálysértések.
- Részletesebb tájékoztatás a főoldalon.
- Új hotspot widget-ek.

A szoftvert támogató termékek, folyamatok, és résztvevő munkatársak periodikus tesztelése és értékelése szükséges ahhoz, hogy rálátásunk legyen a tervezett, implementált és telepített biztonsági kontrollok hatékonyságára. Ez a fő célja a folyamatos monitorozó tevékenységeknek. Ezen felül a folyamatos monitorozás segít meghatározni a tervezett vagy váratlan tevékenységek hatásait a szoftver megbízhatóságára. Továbbá nagy segítséget nyújthat a folyamatos monitorozás a véletlen vagy szándékos támadások felismerésében, melyek a biztonsági réseket használják ki. A monitorozás eszközei közé tartozik a szkennelés (biztonsági rések, hálózat, operációs rendszer), penetrációs tesztelés és behatolásjelző rendszerek használata, melyek mind segítenek az ellátási láncban részt vevő szoftverek és működési környezet figyelésében.

A penetrációs tesztelésnél ügyelni kell arra, hogy mind a külső, mind a belső támadási felületek tesztelve legyenek.

A behatolásjelző rendszerek beállításait érdemes úgy elvégezni, hogy ne csak a szoftveren történő behatolásokat észlelje, hanem a forráskódon történőeket is. Mikor a mintaillesztő behatolásjelző rendszereket használjuk, a folyamatos monitorozó folyamatnak először importálnia kell a legújabb aláírásokat (signatures) kanonikus és nem-kanonikus formájukban egyaránt, valamint konfigurálnia kell őket az ellenőrzés egy részeként. Ez a típusú behatolási rendszer malware behatolásokat is vizsgál. Mikor a viselkedés-behatolási rendszerek anomáliákat fedeznek fel, a monitorozó folyamatnak validálnia kell a szoftvert a normális viselkedés előre beállított, szigorúan ellenőrzött alapjaival szemben.

A hálózati forgalom monitorozásával szintén malware behatolások fedezhetőek fel, főleg abban az esetben, mikor a portoknak és protokolloknak nem kellene nyitva lenniük, mégis kommunikációs tevékenységek fedezhetők fel rajtuk.

***Fogalmak a 5.4. fejezetben:*** termékkockázat, defektus, lefedettség, agilis fejlesztés, penetrációs tesztelés.

# 6. SPECIFIKÁCIÓ ALAPÚ TESZTELÉS

## 6.1 Black-box tesztelés

A black-box, más néven feketedoboz vagy specifikáció alapú tesztelés a tesztelendő programról készült specifikációt vagy azáltal használt modellt veszi alapul, így csak a program működésének eredményét ismerhetjük, annak miként jét nem. A specifikáció nemcsak funkcionális, hanem nem-funkcionális követelményeket is tartalmazhat, így a feketedoboz tesztelés során a használhatóság, a teljesítmény és a biztonság is ellenőrizhető.

Feketedoboz alapú technikák a következők:

- Ekvivalencia partíciók
- Határérték analízis
- Döntési tábla teszt
- Állapotátmenet teszt
- Használati eset teszt
- Kombinációs technikák

### 6.1.1 Ekvivalencia partíciók

Az ekvivalencia partíciók számolásának lényege, hogy az adatokat valamilyen szempontból csoportosítjuk, majd az egyes csoportba sorolt adatokat egyformán kezeljük tesztelés szempontjából. Ezek a csoportok, partíciók tartalmazhatnak érvényes vagy érvénytelen adatokat, így megkülönböztethetünk valid és invalid partíciókat.

A bemeneti partíciókra most mutatunk egy példát:

Tegyük fel, hogy egy program a -10000 és 10000 közötti egész számokkal dolgozik, amely ugyanúgy működik az összes érvényes pozitív és negatív számra.

Érvényes input partíciók:

- -10000 és -1 között
- 0
- 1 és 10000 között

A 0 mint partíciót azért kellett felvenni, mert a specifikáció nem mond róla semmit, így külön kell kezelni.

Mivel a specifikáció szerint a program csak a -10000 és 10000 közötti számokat kezeli, az érvénytelen partíciók a következők:

- -10000 alatti számok
- a 10000 feletti számok
- valós értékek
- betűt tartalmazó sztring

Ezen információk birtokában egy valid és egy invalid teszteset:

- -2000, 0, 2000
- -11000, 11000, 4.56443, szigma

Biztonsági szempontból nem túl jó megoldás az ekvivalencia particionálás használata, mert a rosszindulatú programok, felhasználók felhasználhatják a particionálás adta információkat, valamint bizonyos rendszereknél a határértékek ismerete komoly kockázatot jelenthet.

### 6.1.2 Határérték analízis

A következő, specializáció alapú technika a határérték analízis. A legtöbb hiba általában a határértékek környékén fedezhető fel, ez gyakran köthető bizonyos programozási szerkezetekhez. A vizsgált határértékek lehetnek érvényesek, érvénytelenek, vagy úgynevezett belső szomszédok. Gyakran használják az ekvivalencia partíciók kiegészítéseként. A határértékek meghatározásának fontos az input pontossága, pl. valós értékeknél a határérték mindkét oldalon a vele szomszédos értékeket is tesztelni kell. Vegyünk néhány példát!

1.) A víz forráspontja 100 celsius fok. Ha 1 celsius fok pontossággal szeretnénk meghatározni a határértékeket, akkor a 99, 100 és 101 celsius fokot kell vizsgálni. Ha 0,1 celsius fok pontossággal mérünk, a vizsgálandó értékek a 99,9, a 100 és a 100,1 celsius fok.

2.) Egy 100 pontos vizsgán az értékelések határai: 50, 65, 80 és 90 pont. Így a vizsgálandó értékek a következők:
   – 49, 50, 51
   – 64, 65, 66
   – 79, 80, 81
   – 89, 90, 91

Biztonsági szempontból a használata hasonló veszélyeket rejt, mint az ekvivalencia particionálás – a határértékek kiszivárgása mint fontos információ rosszindulatú felhasználók kezébe kerülve veszélyt jelenthet.

## 6.1.3 Döntési tábla teszt

Szintén feketedoboz technika a döntési tábla teszt, amely az összes, leginkább üzleti döntéseket leíró bemeneti és kimeneti eredményt tartalmazza.

A döntési tábla sorai a döntéseket és az eredményeket definiálják, míg az oszlopok az egyes üzleti szabályoknak felelnek meg. Mivel az összes döntést lefedni igen körülményes és hosszadalmas lehet, a limitált döntési tábla használata terjedt el. Nézzünk egy példát: egy bevásárlóközpontban a kártyatulajdonosok gyűjthetnek pontokat, hogy kedvezménnyel vásárolhassanak. Kártya nélkül csak 100 euró feletti vásárlás esetén van kedvezmény.

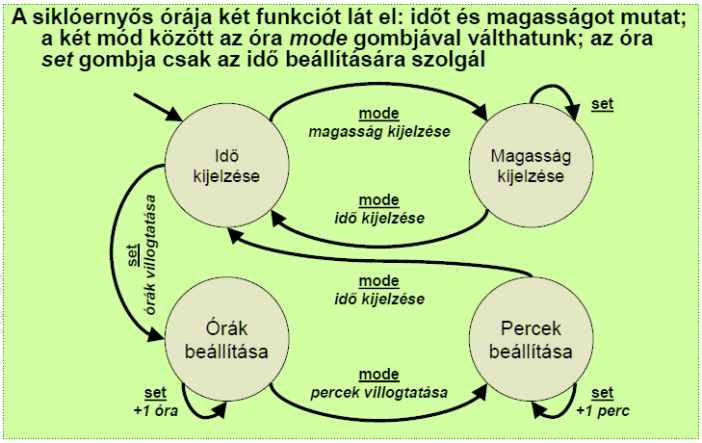
Egy bevásárlóközpontban a kártyatulajdonosok gyűjthetnek pontokat, hogy kedvezménnyel vásárolhatnak. Kártya nélkül csak 100 euró feletti vásárlás esetén van kedvezmény.

| | #1 | #2 | #3 | #4 |
|---|---|---|---|---|
| ***Döntések*** | | | | |
| Van kártya | H | H | I | I |
| Kedvezmény választása | - | - | I | H |
| 100 euró feletti vásárlás | H | I | - | - |
| ***Eredmények*** | | | | |
| Van kedvezmény | H | I | I | H |
| Pontgyűjtés | H | H | H | I |

Biztonsági szempontból hasznos lehet a hiányos specifikáció kiszűrésére, ugyanis döntési tábla teszt felvázolásával az összes, vagy legalább a legfontosabb üzleti döntéseket lefedhetjük.

## 6.1.4 Állapotátmenet teszt

Az állapotátmenet teszt nagyban hasonlít a döntési táblával végrehajtott teszthez. Olyan rendszereken szokás használni, ahol az eredményeket a rendszer állapotában történő változás vagy bemeneti feltételek váltják ki. Az állapotátmenet tesztet egy diagramon vagy táblázatban szokás ábrázolni, ahol az állapotokat körrel, a lehetséges átmeneteket nyíllal jelölik. Az eseményeket ábrázoló nyilakat meg lehet címkézni.




*22. ábra Példa állapotátmenet tesztre*

Az állapotátmenet teszt biztonsági tesztelésre is használható, ugyanis a specifikációban meghatározott tényezőket ábrázolva rájöhetünk műveleti hiányosságokra, így a biztonsági kockázatok is kiszűrhetők.

### 6.1.5 Használati eset teszt

A használati eset teszt szintén fekete doboz technika, ahol a rendszer funkcionalitását különböző felhasználók szemszögéből teszteljük. Egy használati eset egy adott típusú felhasználó és a rendszer között történő interakciót írja le. Minden használati esetnek vannak elő- és utófeltételei, melyeknek teljesülnie kell a sikeres végrehajtáshoz. Ezen felül egy használati eset rendelkezik fő és alternatív ágakkal, amelyek a lehetséges eseményeket és funkciókat írják le. Általában diagrammal ábrázolják, melynek részei a rendszer, valamint az azon kívüli és azon belüli szereplők és események (lásd a 7. ábrát).

Biztonsági szempontból a use case tesztelés használható olyan formában, hogy szerkezeti optimalizálás elemzést végezzen a biztonsági mechanizmusokon a rendszer különböző pontjain, így létrehozva egy dokumentumot az optimális döntéseknek. Továbbá a szereplőket a use case diagramot elemezve felépíthetünk egy képet a szereplő jogosultsági hierarchiájáról, amelyeket szerepkörönként és csoportokként ábrázolhatunk a diagramon.

### 6.1.6 Kombinációs technikák

Amint azt már a 3-as fejezetben is említettük, kombinációs technikákat akkor használunk, mikor számos paraméter szükséges a szoftver teszteléséhez, ahol az egyes paramétereknek is több értéke lehet, így a szükséges kombinációk száma igencsak megnövekszik.

### 6.1.7 Tapasztalati alapú technikák

Tapasztalati alapú technikákat akkor is használhatunk, amikor valamilyen ok miatt nem tudunk szisztematikus tesztelést végezni, pl. tesztesetek készítésére alkalmatlan specifikáció esetén, vagy ha csak kevés időnk van a tesztelés elvégzésére. Ezen felül strukturált tesztelés mellett is érdemes elvégezni. A tapasztalat alapú technikák lényege, hogy a tesztelő az eddig szerzett tapasztalatait és tudását használva próbálja felderíteni a rendszer legkritikusabb részeit.

Ilyen technika a hibasejtés is, amelyet akkor használunk, mikor módszeres tesztekkel nehezen felfedezhető hibákat, vagy ismert és várható hibákat kell detektálni. Hátránya, hogy a módszer eredményessége erősen függ a tesztelő tapasztalataitól.

A felderítő tesztelés a tesztelő tapasztalatát a módszerességgel egyesíti. Mind hiányzó vagy rossz specifikáció, mind időhiány esetén hasznos módszer. Főbb jellemzői közé tartozik, hogy a tesztelési célok köré szerveződik, hogy időkeretekkel dolgozik, valamint hogy párhuzamosan történik a tesztek tervezése, végrehajtása, kiértékelése, és a folyamatos „tanulás”.

***Fogalmak a 6.1. fejezetben:*** specifikáció alapú tesztelés, ekvivalencia partíciók, határérték analízis, döntési tábla teszt, állapotátmenet teszt, használati eset teszt.

## 6.2 Tapasztalat alapú tervezés

[SRC: http://people.cs.aau.dk/~bnielsen/TOV07/lektioner/blackbox-07.pdf]
[SRC: http://en.wikipedia.org/wiki/Black-box_testing]

Az előző fejezetben felsorolt teszt tervezési technikákon kívül léteznek tapasztalat-alapú módszerek, amelyek szintén feketedoboz tesztelésnél használhatóak:

- hibasejtés (error guessing)
- véletlenszerű teszt (random testing)
- ellenőrző lista alapú teszt (checklist-based testing)
- felderítő tesztelés (exploratory testing)

### 6.2.1 Hibasejtés

A hibasejtés szintén egy feketedoboz technika, ahol csak sejthető, hogy hol található hiba a szoftverben, így a tesztelő csak az intuícióit és tapasztalatait tudja használni. Valójában nem is igazán technikaként kezeljük a hibasejtést, hanem inkább egy ad-hoc módszernek nevezzük. A hibasejtés stratégiája igen egyszerű: készítsünk egy listát a lehetséges hibákat előidéző szituációkról, majd ezen listára alapulva írjunk teszteseteket. Tipikus hibák lehetnek pl. a 0-val való osztás, invalid paraméterek, üres input, üres fájlok vagy hibás formátumú adatok használata.

Ha egy kicsit szofisztikáltabban kezeljük a hibasejtést, akkor már kockázatelemzésről beszélünk.

A hibasejtés ezen felül igen időtakarékos tesztelési módszer, mivel a feltételezéseket és tippeket tapasztalt tesztelők hozzák meg, így olyan hibák is könnyen megtalálhatóak, melyeket a szisztematikus tesztelés nem mutat ki.

A hibasejtés sikeressége nagyban függ a tesztelők tapasztalatától és tudásától, ezért sokszor csak a szisztematikus tesztelés után használják.

### 6.2.2 Véletlenszerű tesztelés

A véletlenszerű tesztelés alapötlete, hogy véletlenszerűen vezessük a rendszert tipikus, extrém vagy ritka szituációkba. A véletlenszerű tesztelés növeli a rendszerhibák megtalálásának esélyét. Több területen is alkalmazzák:

- olyan rendszereknél, melyek folyamatosan futnak nem determinisztikus módon, ilyenek pl. a vezérlő vagy kommunikációs rendszerek
- olyan rendszereknél, melyek hatalmas bemeneti tartománnyal rendelkeznek.

Példák a véletlenszerű tesztelés használatára:

- véletlenszerű egérkattintások a grafikus felületen,
- tipikus böngészési viselkedés (pl. kattintás, olvasás) egy tipikus, véletlenszerű várakozási idővel,
- a specifikációs állapotmodell véletlenszerű körbejárása tesztelés alatt.

### 6.2.3 Ellenőrző lista alapú tesztelés

Az ellenőrző lista alapú teszt egy olyan tapasztalat alapú teszttervezési technika, amely során a tapasztalt tesztelő magas szintű tesztelemeket jegyez fel, ellenőriz, vagy szabályokat és kritériumokat használ a program verifikálásához.

A lista tartalmazhat tesztelendő tételeket, szabályok listáját vagy bizonyos ellenőrizendő kritériumokat, feltételeket. Ezeket a listákat általában az idő múlásával a tesztelők fejlesztik, mind a fejlesztők tapasztalatait, mind szabványokat, korábbi problémás területeket és ismert használati forgatókönyveket alapul véve. A lefedettséget a lista befejezettsége határozza meg. Az ellenőrző lista alapú tesztelés egy könnyű, gyors módja a tesztelésnek, miközben a lefedettségről szerzett információk folyamatosan karbantarthatóak.

### 6.2.4 Felderítő tesztelés

A felderítő tesztelés, ahogy a nevéből is sejthető, felfedezi, valamint megpróbálja kitalálni, mit csinál a szoftver, mit nem csinál, mi működik és mi nem. A tesztelő folyamatosan döntéseket hoz arról, mi legyen a következő tesztelés célja, figyelembe véve a rendelkezésre álló erőforrásokat. Akkor tudjuk a legjobban alkalmazni, mikor a specifikáció erősen hiányos, vagy csak nagyon rövid idő áll rendelkezésünkre.

A tervezés magában foglalja a teszt fejezetek létrehozását, egy rövid deklarációt a tesztelés körének meghatározására (ez általában 1-2 óra), a célokat és a lehetséges megközelítéseket.

A teszttervezési és futtatási tevékenységeket a tesztelők párhuzamosan hajtják végre anélkül, hogy hivatalosan dokumentálnák a tesztfeltételeket, teszteseteket vagy tesztszkripteket. Ez természetesen nem jelenti azt, hogy egyáltalán nem történik formális tesztelés. Például a tesztelő dönthet úgy, hogy hivatalosan, dokumentáltan végez pl. határérték-analízist, de az elemzést megteheti az értékek rögzítése nélkül is. Néhány jegyzet készül a felderítő tesztelés során, így a riport így is elkészíthető a későbbiekben.

***Fogalmak a 6.2. fejezetben:*** hibasejtés, véletlenszerű teszt, kockázatelemzés, ellenőrző lista alapú teszt, felderítő tesztelés.

## 6.3 Teszt tervezési technikák

A teszt tervezési technikáknak 4 fő típusa van:

- Specifikáció alapú technikák (5. fejezet)
- Struktúra alapú technikák (6. fejezet)
- Modell alapú technikák (jelen fejezetben beszélünk róla)
- Gyakorlat alapú technikák (5.2 fejezet)

A feketedoboz teszttervezési technikát az ekvivalencia partíciókon, a határérték-analízisen, az állapotátmenet teszten, a használati eset teszten, a kombinációs technikákon és a tapasztalat-alapú tesztelésen kívül még az alábbi módokon valósíthatjuk meg:

- ok-okozat diagram
- stressz teszt
- all-pairs teszt

### 6.3.1 Ok-okozati diagram

Az ok-okozat diagram bemeneti feltételek kombinációját teszteli, ahol az ok a bemenetekből és az aktuális állapotból, az okozat a kimenetekből és az új állapotból áll. Ehhez készíthető egy boolean típusú gráf, amely összeköti az okokat és okozatokat. Ezen felül feljegyzi azokat a kombinációkat, amelyeket lehetetlen előidézni, majd egy döntési táblában szemlélteti a gráfban leírtakat, amelyből később tesztesetek készíthetők.

Tehát az ok-okozat diagram egy szisztematikus módszer, amely segítségével a feltételek kombinációiból tesztesetek készíthetők. Mivel az összes kombináció ábrázolása majdhogynem lehetetlen, vagy igen nagy adathalmazt generál, olyan heurisztikus módszerek is használhatóak, melyek csökkentik a teszteléshez szükséges kombinációk számát.

### 6.3.2 Stressz teszt

A stressz teszt, más néven teljesítmény vagy terhelés teszt lényege, hogy figyeli a rendszer viselkedését a teljesítmény szélsőséges eseteiben. Példa a terhelés tesztelésére:

- magas számú felhasználó beléptetése a tesztelendő alkalmazásba,
- a szolgáltatás támadásainak tagadása, figyelmen kívül hagyása,
- mosógép kikapcsolása működés közben egy tetszőleges állapotban,
- nagymértékű kérés küldése hálózati kapcsolaton minél rövidebb idő alatt.

Az all-pairs nevű tesztelési technika a lehetséges inputok páros kombinációjával tesztel. Ezt a tesztelési típust számos eszköz segíti.

### 6.3.3 Modell alapú teszttervezési technikák

Míg a fent említett technikák a specifikáció alapú teszttervezési technikákhoz tartoztak, addig a következőkben a modell alapú tesztelési technikákkal fogunk megismerkedni.

A modell alapú tesztelési technikák (model-driven testing) valójában a specifikáció alapú technikákhoz tartoznak, csak annál formalizáltabb módszerekkel dolgoznak. Közvetlenül az UML modellből vezeti le a teszteseteket, és formalizált teszt specifikációt alkalmaz. Erre használható az UML kiterjesztése (UTP – UML Testing Profile).

A modell alapú tesztelés a modell-alapú tervezés alkalmazása a szoftvertesztelés kivitelezésére. A modelleket ezen felül használhatjuk a rendszer alatti kívánt viselkedés, vagy tesztstratégiák és tesztkörnyezet reprezentálására.

A rendszer alatti viselkedést leíró modell gyakran egy absztrakt, részleges bemutatása a kívánt viselkedésnek. Ebből az absztrakt leírásból készíthetünk absztrakt teszteseteket, amelyek még nem használhatók éles futtatási környezetben. Ezért az absztrakt tesztesetekből futtatható teszteseteket kell létrehozni, amelyek közvetlenül tudnak kapcsolatot teremteni a rendszerrel és annak viselkedésével. Ezt elérhetjük az absztrakt tesztesetek leképezésével.

Néhány modell-alapú tesztelési környezetben a modell már tartalmaz annyi információt, hogy közvetlenül generálhatóak legyenek a futtatható tesztesetek. Ekkor az absztrakt teszthalmaz csak fogalmi szinten szerepel a folyamatban, nem mint konkrét teszthalmaz.

Tesztesetek létrehozása modellekből többféleképpen történhet. Mivel a tesztelést gyakran tapasztalati alapon heurisztikákkal számolva hajtják végre, nem ismert egy olyan módszer sem, amely biztosan a legjobb megközelítés lenne a szoftvertesztelés kivitelezéséhez. Viszont a minél jobb tesztelési eredmény eléréséhez összegyűjthetjük a követelményeket, használati eseteket, és egyéb információkat, melyeket gyakran a használt modell ismeretének segítségével nyerhetünk.

***Fogalmak a 6.3. fejezetben:*** ok-okozat diagram, kockázatelemzés, stressz teszt, all pairs tesztelés, modell alapú tesztelés.

## 6.4 Biztonsági teszt típusok

A biztonsági tesztelés több dimenzióban vizsgálja a rendszer biztonságát:

- bizalmasság,
- integritás
- hitelesítés,
- elérhetőség,
- jogosultságok,
- letagadhatatlanság (non-repudiation)

Minden egyes dimenziót más és más szempont alapján teszteljük, így ezek felfoghatók biztonsági tesztek külön típusaiként is.

A **bizalmasság** olyan tényező, amely véd az információk közzétételével szemben.

Az **integritás** célja, hogy a kapó fél számára lehetővé tegye, hogy megbecsülhesse, hogy a kapott információ helyes-e. Az integritás sémák gyakran használnak a bizalmassági sémákkal hasonló technológiákat, de az előbbi plusz információkkal dolgozik.

A **hitelesítés** lényege a felhasználók azonosítása, valamint annak a meghatározása, hogy a termék valóban a kívánt termék-e, és az megbízhatóan működik-e.

Az **elérhetőség** tényező biztosítja a kellő információt és kommunikációs csatornákat, amelyek bármikor készen kell álljanak a használatra a megfelelő felhasználó számára.

A **jogosultság** a biztonság azon dimenziója, amely meghatározza, hogy a felhasználó, információt kérelmező személy jogosult-e a kívánt információkra. A jogosultság egyik példája a hozzáférés szabályozása.

Digitális biztonság témában a letagadhatatlanság azt vizsgálja, hogy az információ eljutott-e a kérelmező személyhez. A letagadhatatlanság egyben azt is biztosítja, hogy az elküldött üzenet később igazolja a küldő és a fogadó felet is.

Kifejezetten struktúra alapú biztonsági tesztelés technikák a következők lehetnek:

- fuzzing
- szintax tesztelés
- felfedező tesztelés
- adatelemzés
- teszt állványozás (? test scaffolding)
- program viselkedésének analizálása

## 6.4.1 Fuzzing

A fuzzing technika lényege program interfészek megzavarása hozzáadott zajokkal. Például a tesztelő vagy a tesztelő eszköz a szoftver által generált rendszerhívásokat kap el, melyek egy fájl olvasása közben jöttek létre, majd ezekbe véletlen sorrendben generált bájtokat szúr.

Tegyük fel, hogy a tesztelő kicserélte a szoftver által olvasott fájlt egy másik, random biteket tartalmazóra. Ha a rendszer ennek eredményeképpen összeomlik, valószínűsíthető, hogy a szükséges ellenőrzés kimaradt a programból, így a helytelen formátumú és tartalmú fájlokat is elfogadja. Az ellenőrzés hiányát könnyen kihasználhatják a rossz szándékú felhasználók, kár okozva a rendszerben.

## 6.4.2 Szintaxis tesztelése

Ez a fajta feketedoboz biztonsági tesztelés technika a bemeneti értékek szintaktikai specifikációján alapul. Például a program tesztelhető úgy, hogy inputnak hibás, vagy üres értékeket adunk, esetleg felcseréljük a bemeneti változókat, stb. Gyakori, hogy egy webalapú alkalmazás használata során a behelyettesített érték egy HTTP kérés metakarakterekkel vagy javaScript kóddal, amelyet számos esetben szűrni kellene. Továbbá a puffer túlcsordulása is ily módon tesztelhető, hosszú input sztringek megadásával.

Így a szintakszis tesztelés segít a tesztelőknek eldönteni, hogy a program a bemeneti értékeket helyesen ellenőrzi-e, avagy megtörténik-e egyáltalán az ellenőrzés.

## 6.4.3 Felfedező tesztelés

Biztonság tesztelése esetén hasznos lehet olyan teszteket is végrehajtani, amelyekkel ismeretlen kimenetelű funkciókat tesztelnek.

Az alapötlet az, hogy a tesztelő így olyan anomáliákat is feltárhat, amelyeket a szoftverproblémák okozójához juttatnak el.

Nincs technikai oka annak, hogy a teszterv miért nem tudja ezeket a teszteseteket lefedni, de a gyakorlatban számos tesztelő találja hasznosnak a felfedező tesztelést. Az ezen tesztelés által felfedezett finom anomáliák rávezethetik a tesztelőt további információk gyűjtésére, újabb tesztesetek létrehozására, amely így a teljes tesztelést segítheti.

## 6.4.4 Adatelemzés

Az adatelemzés lényege, hogy a tesztelők megpróbálnak újabb információkat szerezni a program belső működéséről az általa generált kimenetek alapján. Az adatelemzés két fontos aspektusa:

A státusz nélküli protokollok külső mechanizmusokat használnak arra, hogy egy tranzakció állapotát nyomon kövessék, mint pl. ahogy a HTTP sütiket használ erre. Nem túl kellemes, ha ezek az információk hackerek kezébe kerülnek, de adatelemzéssel a státuszinformációk meghatározhatóak.

### 6.4.5 Program viselkedésének analizálása

A program viselkedésének folyamatos figyelése minden tesztelési módszer számára kulcsfontosságú feladat. A viselkedés figyelésével megállapítható, hogy vannak-e biztonsági rések a szoftverben. A figyelés során a tesztelő nem meghatározott tüneteket keres, amely váratlan biztonsági hibákra figyelmeztethet.

### 6.4.6 Test Scaffolding

Ezalatt a tesztelési forma alatt olyan eszközöket értünk, melyek támogatják a tesztelő tevékenységét pl. tesztadatok generálásával.

***Fogalmak a 6.4. fejezetben:*** fuzzing, szintaxis tesztelés, felfedező tesztelés, adatelemzés, monitorozás.

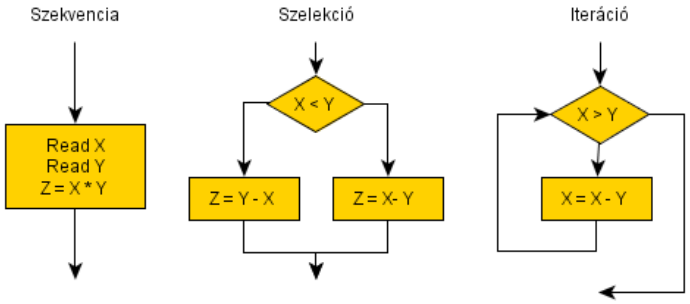
# 7. STRUKTÚRA ALAPÚ TESZTELÉS

## 7.1 White-box tesztelés

A fehérdoboz tesztelés a program szerkezetének felderítésére szolgál. Mivel a helyes működés vizsgálata leginkább a feketedoboz tesztelés feladata, így a fehérdoboz tesztelés a teljes kód ellenőrzését végzi.

A fehérdoboz tesztelés több szinten is vizsgálhatja a kódot:

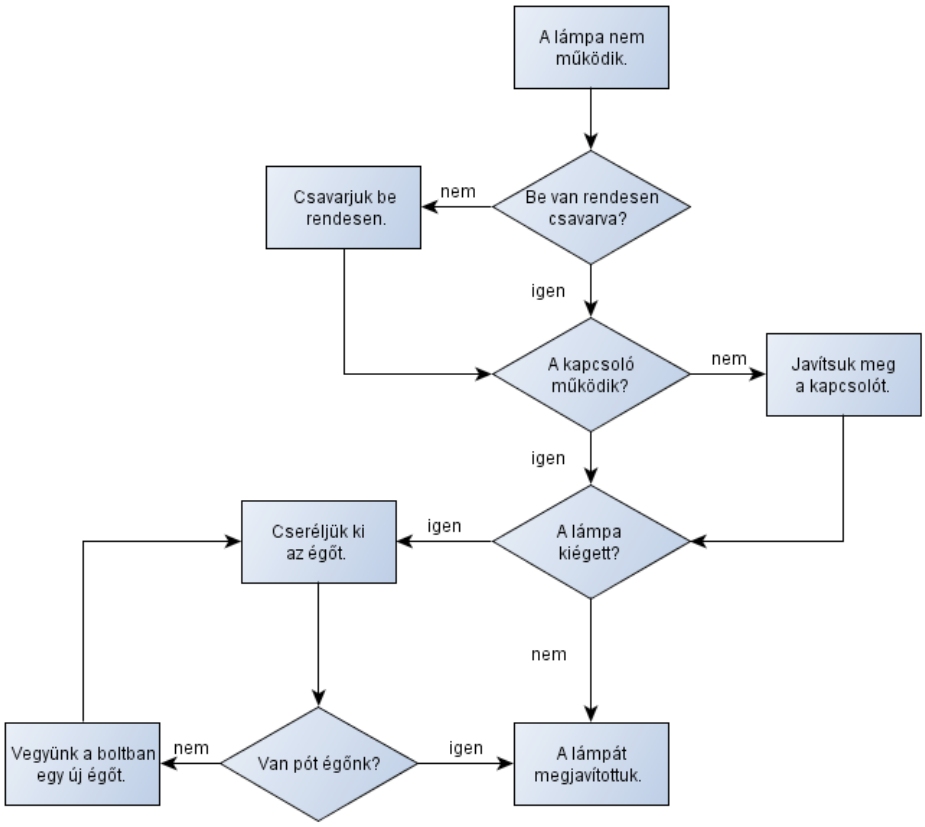
- programozási struktúrákat komponens szinten, ilyenek pl. a szekvencia, szelekció és iteráció,
- hívási függőségeket iterációs szinten,
- egyéb szerkezeti elemeket rendszer szinten, ilyen lehet pl. a menü, egy üzleti folyamat vagy weboldal.




*23. ábra A szekvencia, a szelekció és az iteráció*

A program-struktúra reprezentációját háromféleképpen tehetik meg a tesztelők, méghozzá pszeudokóddal, folyamatábrával vagy vezérlési folyam gráffal.

A folyamatábra a program működésének vizualizálására szolgál, ahol az utasításokat egy téglalappal, míg a döntéseket rombusszal jelölik.




*24. ábra Példa folyamatábrára*

Ezzel szemben a vezérlési folyam gráf a tesztelendő szoftver döntési folyamatának modellezésére szolgál, amely csak azokat az elemeket jeleníti meg, melyek az adott vezérlési folyamra hatással vannak. A vezérlési folyam gráf hierarchikus is lehet, ahol részgráfok rajzolhatók kisebb kódrészletekre. A vezérlési folyam gráfot három lépésben építhetjük fel:

- vezérlési struktúrák azonosítása azokkal az utasításokkal, amelyek hatással vannak a vezérlési folyamra,
- döntésekhez gráfpont hozzárendelése,
- gráfpontok kifejtése a vezérlési struktúrának megfelelően.

### 7.1.1 Biztonsági fehérdoboz tesztelés

Mint a feketedoboz tesztelés esetében is, fehérdoboz tesztelést is alkalmaznak a biztonság ellenőrzéséhez.

A fehérdoboz alapú biztonsági tesztelést egy olyan megközelítésként lehet definiálni, amely feladata a kód ellenőrzése biztonsági rések feltárása céljából. A fehérdoboz biztonsági tesztelés alanyai a következők lehetnek:

- kód,
- biztonsági követelmények,
- tervezési dokumentumok,
- architekturális és tervezési kockázatelemzés dokumentum.

A fentiek mellett fontos ügyelni arra, hogy ismerjük a minőségi követelményeket a teljesítményre nézve, így a biztonság a szoftverhez igazítottan az optimális szintre hozható. Ez különösen fontos webalapú rendszereknél, ahol a teljesítményt nagyban befolyásolhatja a szoftver biztonsága. A kockázatelemzési dokumentum segít felfedezni a fenyegetéseket és sérülési lehetőségeket, azok bekövetkezési valószínűségét és hatásait az egyes komponensekben.

A kimerítő tesztelés majdhogynem az összes rendszer esetében lehetetlen, valamint nem túl költséghatékony. A tesztelendő programrészek, lehetséges inputok kiválasztásában nagy segítséget nyújt a kockázatelemzési dokumentum, amely a lehetséges teszteseteket priorizálja úgy, hogy a kockázatosabb programrészek tesztelését előre veszi a rangsorban. A teszt stratégiát ebből kifolyólag úgy érdemes tervezni, hogy meghatározunk egy optimális lefedettségi szintet, amely időben és erőforrásokban is kivitelezhető. Mint az általános tesztelésnél is, úgy a biztonsági tesztelés teszt stratégiájában is meg kell határozni a tesztelés hatókörét, a lefedettségi metrikákat, a tesztelésben szereplő munkatársakat és szerepkörüket, valamint a tesztkörnyezetet. Mivel a biztonság tesztelése igen fontos, főleg üzletkritikus vagy értékes adatokat tároló rendszerek esetében, a teszt stratégia kidolgozásánál törekedni kell a tesztelés hatékonyságára és eredményességére.

A struktúra alapú tesztelést már a szoftverfejlesztési folyamat korai szakaszában el tudjuk kezdeni, így a biztonság mint tényező már beépíthető az alkalmazásba. Ez a típusú teszt tervezési technika segít a fejlesztőknek abban, hogy elég rugalmas kódot írjanak ahhoz, hogy a biztonsági rések könnyen felismerhetőek legyenek. Azonban a fehérdoboz tesztelés nem fedi le a kódbeli függőségeket (szolgáltatások, könyvtárak stb.) és a külső komponenseket. Egy kis betekintést nyújt a biztonsági rések kihasználhatóságába, habár ezzel még nem biztos, hogy pontos képet kapunk a lehetséges kockázatokról, így a tesztelés sikeréhez feketedoboz technikák használatára is szükség van.

Fehérdoboz tesztelésen belül megkülönböztetünk különböző technikákat, ezek a

- utasítás teszt és lefedettség,
- döntési teszt és lefedettség,
- vagy egyéb technikák.

Erről a következő fejezetben lesz szó.

***Fogalmak a 7.1. fejezetben:*** fehérdoboz tesztelés, szekvencia, szelekció, iteráció, pszeudokód, folyamatábra, vezérlési folyam gráf, kimerítő tesztelés, utasítás lefedettség, döntés lefedettség, utasítás teszt, döntési teszt.

## 7.2 Lefedettség

Amint azt az előző fejezetben is említettük, fehérdoboz tesztelésen belül megkülönböztetünk különböző módszereket, ezek

- az utasítás teszt és lefedettség,
- a döntési teszt és lefedettség,
- vagy egyéb technikák.

A lefedettség vagy kódlefedettség megmutatja, hogy a forráskód hány százaléka lett lefedve a tesztelés által. Minél nagyobb a kódlefedettség egy programnál, annál részletesebben lett tesztelve, és annál kisebb a valószínűsége, hogy az bugokat tartalmaz. Ebből kifolyólag a kisebb lefedettséggel rendelkező programok vagy programrészek nagyobb eséllyel hibásodhatnak meg a gyérebb tesztelésből adódóan.

## 7.2.1 Utasítás teszt és lefedettség

A lefedettség típusok közül az utasítás teszt lefedettség célja az utasítások módszeres vizsgálata. A kapott lefedettségi százalék az átvizsgált és az összes végrehajtható utasítás aránya. A lefedettség meghatározására és ábrázolására folyamatábra és hibrid vezérlési folyam gráf alkalmas. Nézzünk egy példát!

```
PROGRAM LefedettsegPelda
A, B: INTEGER
BEGIN
    READ A
    READ B
    IF A > 1 AND B = 2
          THEN B = B / A
    ENDIF
    IF A = 2 OR B = 2
          THEN B = B + 1
    ENDIF
END
```

A példában a 100%-os lefedettség egyetlen tesztesettel lefedhető: A = 2 és B = 2.
Mivel a fenti program elágazásokat is tartalmaz, az utasítás-lefedettség nem elégséges.

## 7.2.2 Döntési teszt és lefedettség

Hogy javítsunk a lefedettségen, ismerjük meg a döntési teszt lefedettséget!

A döntési teszt célja a döntések módszeres átvizsgálása. Fontos, hogy minden döntési pontnál mindkét ágat legalább egyszer érintsük! A döntési lefedettség a minden kimeneti ágon tesztelt és az összes döntés arányát jelenti. Ábrázolásához (hibrid) vezérlési folyam gráf a legmegfelelőbb.

```
PROGRAM Check
Count, Sum, Index: INTEGER
BEGIN
    Index = 0
    Sum = 0
    READ Count
    READ New
    WHILE Index <= Count
    DO
          IF New < 0
          THEN
                Sum = Sum + 1
          ENDIF
          Index = Index + 1
          READ New
    ENDDO
    PRINT(Sum, „negatív számok”)
END
```

A While miatt elegendő egyetlen teszteset a 100%-os döntési lefedettséghez, pl. 2, -1, 1.

Nézzünk egy másik példát, ahol már nem elég egy teszteset a 100%-os lefedettséghez.

```
PROGRAM AgeCheck
Candidate: Integer
BEGIN
    READ CandidateAge
    IF CandidateAge <18
          THEN PRINT(„Fiatal”)
    ELSE
          IF CandidateAge > 30
                THEN PRINT(„Idős”)
          ELSE PRINT(„Megfelelő életkor”)
          ENDIF
    ENDIF
END
```

Ebben az esetben a 100%-os lefedettséghez három tesztesetre van szükség:
- 16
- 21
- 40

Több egyéb technika is létezik, amelyek különböző lefedettségeket mérnek. Jelen dokumentumban hármat mutatunk be:
- Feltétel lefedettség
- Többszörös feltétel lefedettség
- Útvonal lefedettség

## 7.2.3 Feltétel lefedettség

A feltétel lefedettség az egy döntésen belüli minden elemi részfeltétel igaz/hamis kimenetének ellenőrzésén alapuló lefedettség, míg a többszörös feltétel lefedettség egy döntésen belül az elemi részfeltételek minden kombinációját ellenőrzi.

A feltétel lefedettséget predikátum lefedettségként is szokás emlegetni, ahol minden egyes igaz/hamis kifejezést végrehajtunk úgy, hogy azok legalább egyszer igazak és hamisak legyenek. Nézzünk egy példát:

```
if ((A || B) && C)
{
  /* Utasítások */
}
else
{
  /* Utasítások */
}
```

Ahhoz, hogy megbizonyosodjunk a feltétel lefedettség teljességéről, az A, B, és C változókat legalább egyszer végre kell hajtani igaz és hamis értékkel. Így a fenti példát tekintve elég 3 teszteset ahhoz, hogy 100%-os lefedettséget érjünk el:
- A = igaz, B = nincs értékelve, C = hamis
- A = hamis, B = igaz, C = igaz
- A = hamis, B = hamis, C = nincs értékelve

## 7.2.4 Útvonal lefedettség

Az útvonal lefedettség feladata az összes lehetséges végrehajtási útvonal ellenőrzése legalább egyszer. Minden lehetséges útvonalat be kell járnunk, beleértve a hurkokat 0-szor, 1-szer és többször végrehajtva.

Vannak olyan hibák, melyeket a feltétel lefedettséggel nem tudunk felfedezni, ilyet tartalmaz a következő kód:

```
$h = 0;
if ($x)
{
    $h = { a => 1 };
}
if ($y)
{
    print $h->{a};
}
```

A 100%-os feltétel lefedettség elérhető az x és y változók 1-1 és 0-0 értékekre állítással, viszont az x = 0 és y = 1 esetében a futtatás sikertelen lesz.

A fenti példát tekintve négy útvonal elég ahhoz, hogy bejárjuk a teljes útvonalat, figyelve arra, hogy a ki nem írt else ágak is egy-egy útvonalnak számítanak.

### 7.2.5 Egyéb lefedettségek

Ezen felül még számos lefedettséget használhatunk, ilyenek a következők:

- Linear Code Sequence and Jump (LCSAJ) lefedettség: minden lineáris kód szekvenciát és hurkot lefuttattunk?
- JJ-út lefedettség: minden útvonalat huroktól hurokig lefuttattunk?
- Be- és kilépési lefedettség: a funkció minden lehetséges hívását és visszatérését lefuttattuk?
- Hurok lefedettség: minden egyes hurkot lefuttattunk 0-szor, 1-szer és többször lefuttatva?
- Állapot lefedettség: minden egyes állapotot lefuttatunk egy véges állapotú gép esetében?

Biztonságkritikus szoftvereknél gyakran fontos a 100%-os lefedettség biztosítása.

Néhány lefedettség szorosan kapcsolódik a másikhoz. Vegyük pl. az útvonal lefedettséget, amely magában foglalja a döntési, utasítási és be- és kilépési lefedettséget. A döntési lefedettség pedig tartalmazza az utasításszintű lefedettséget, mivel minden egyes utasítás egy elágazás része.

A lefedettség nemcsak programrészekre, hanem más szinteken is alkalmazható:

- modulok vagy interfészek lefedettsége
- menürendszer lefedettsége
- üzleti folyamatok lefedettsége

***Fogalmak a 7.2. fejezetben:*** utasítás teszt, utasítás lefedettség, döntési teszt, döntési lefedettség, bug, feltétel lefedettség, többszörös feltétel lefedettség.

## 7.3 Tesztesetek tervezésének segítése

Mivel a fehérdoboz tesztelés a teljes forráskódot veszi górcső alá, így a tesztelést olyan eszközök segíthetik, amelyek a statikus tesztelésen túlmutatva a forráskódot tesztelik a már említett szinteken:

- programozási struktúrákat komponens szinten, ilyenek pl. a szekvencia, szelekció és iteráció,
- hívási függőségeket iterációs szinten,
- egyéb szerkezeti elemek rendszer szinten, ilyen lehet pl. a menü, egy üzleti folyamat vagy weboldal.

Nagyon kevés fehérdoboz teszttervezési tesztechnika oldható meg anélkül, hogy a program módosítása ne legyen szükséges – az értékek megváltoztatásával elérhetünk különböző végrehajtási utakat, vagy a bemeneti lehetőségek teljes halmazát legenerálhatjuk. Hagyományosan ezeket a módosításokat a tesztelők interaktív debuggerekkel vagy manuálisan viszik véghez. Ez a módszer lehetséges, hogy kisebb alkalmazásoknál működik, de nagyobb rendszerek esetében nem alkalmazható. Ennek több oka is van: debuggerek használata nagyban befolyásolja a fejlesztésre, tesztelésre szánt időt, valamint egy nagyobb programnál a forráskód módosítása igen kényelmetlen lehet.

Számos tesztelési eszköz használható fehérdoboz tesztelés végrehajtására anélkül, hogy módosítani kelljen a kódot, vagy tetemes összeget kelljen kiadni egy interaktív debuggerre. Az eszköz használatának előnyei a következők:

– Ezek az eszközök igen gyorsan tesztelnek és debuggolnak, mert nincs szükségük teszttámogató kódra, amelyet be kéne illeszteni. A kereskedelmi szoftverfejlesztésnél számos fejlesztő és önálló osztály felelős a tesztelés és integráció lebonyolításáért, így időt takaríthatunk meg.
– a felhasználás hatékonyságát a tesztkörnyezet maga is javíthatja. Tegyük fel, hogy 747 futó szimulátor különleges hardverbeállításokat igényel. Ha minden tesztelő eszközöl valamilyen változtatást a szimulátoron futó szoftveren, az egyes tesztek konfigurálása sokkal tovább tarthat, valamint az egyes tesztelési forgatókönyvek is több hibát tartalmazhatnak. 747 tesztkörnyezet karbantartása pedig igen költséges lenne.
– A forráskód nem biztos, hogy az összes szoftver számára elérhető. Ezt gyakran úgy szokták megoldani, hogy bevezetnek egy harmadik eszközt, amelyet egy külső cég állított elő. Ezek az eszközök nem szemetelik tele a forráskódot, valamint nem építik újjá azt annak megváltoztatása után.

Jellemzően egy fehérdoboz teszttervezési eszköz egy magas-szintű programozási interfészt biztosít kódírásra, amely segítségével egy speciális funkció performálható vagy egy adott típusú információ nyerhető. További opcióként szerepelhet ilyen eszközökben patch-ok futtatása az alkalmazás részeként maximális sebességen, amelyek segíthetnek az elemzésnél. Továbbá egyes eszközök révén automatikusan torzíthatnak, visszafejthetnek elnevezéseket, így a felhasználók a forráskód elnevezéseket képesek lesznek használni még abban az esetben is, ha a futtatandó kód tartalmaz torzított objektum kód elnevezéseket.

Hogy illusztráljuk a helyzetet, nézzük egy Aprobe példát. Az Aprobe ANSI C nyelvet használ alapnyelvként, néhány kulcsszóval kiegészítve. Ezen a nyelven kerülnek meghatározásra a patch-ok, így az olvasás-írás egyszerűen megvalósítható.

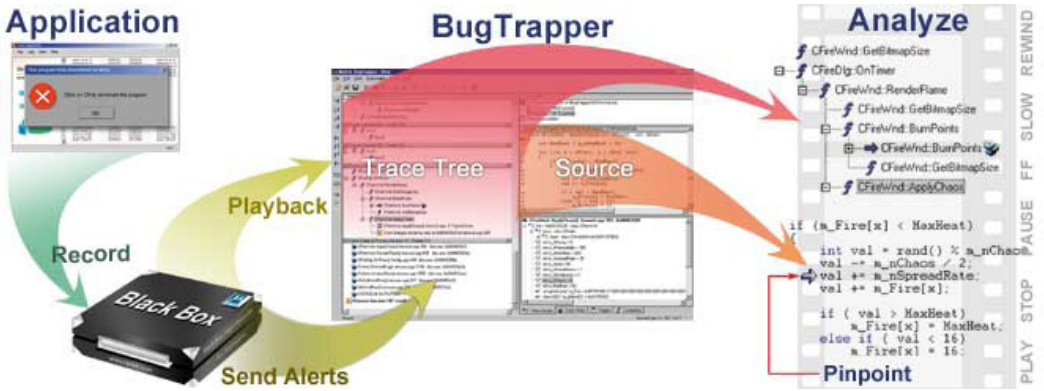
Az aprobe.h header fájl számos funkciót tartalmaz, amelyek hasznosak mind tesztelésnél, debuggolásnál és benchmarkingnál. Az összes ilyen funkció kifejező névvel rendelkezi, és „ap_” karakterekkel kezdődik. Ezek a függvények a tesztelési folyamatot támogatják.

A legtöbb fehérdoboz teszttervezés eszköz valamilyen úton-módon módosítja az eredeti forrást, vagy bizonyos hibák osztályát ellenőrzi. A Rational's Purify például módosítja a lefordított kódot, így minden egyes betöltésnél vagy tárolásnál inicializál egy memóriahely-ellenőrzést, amely eredményét összehasonlítja egy adott kritériummal.

*Forrás: http://www.drdobbs.com/tools/white-box-testing/184404030*

A Rational Purify[4] fő funkciója a memória szivárgások detektálása, mint pl. a nem inicializált memória hozzáférés, puffer túlcsordulása, vagy a memória helytelen felszabadítása.

A BugTrapper[5] nevű program módosítja szintén a futtatandó kódokat a rendszerhívások rögzítése végett. Rekordokat generál arról, hogy éppen mi történik a programban, majd az utolsó pillanatokban, mielőtt a hibás viselkedés megtörténne, megfogja a bugot és annak okát. Lényegében egy távoli szenzorként funkcionál, hogy ne kelljen reprodukálni a felhasználó viselkedését, az adatokat és a környezetet.




*25. ábra A BugTrapper program főbb lépései*[5]

Amint egy bugot elkapott a program, annak elemzése offline is megtörténhet. A fejlesztő megtekintheti a követési információkat anélkül, hogy a bugot reprodukálni kellene.

4 http://www-03.ibm.com/software/products/en/rational-purify-family/
5 http://www.geyra.com/mutek/bugTrapper.htm

A saját fejlesztésű szoftverminőség mérő menedzsment eszköz a QualityGate[6] rendszer, amely a rendszer azonnali, automatikus, objektív elemzését biztosítja, valamint megakadályozza a gyenge minőségű rendszerek befolyását a tesztelt alkalmazásra.

***Fogalmak a 7.3. fejezetben:*** fehérdoboz tesztelés, szekvencia, szelekció, iteráció, debugger, szimulátor, patch.

## 7.4 Lefedettség mérése, monitorozása

A teszt lefedettség a szoftver vagy adott funkció tesztelésére vonatkozik, amelyet többféleképpen határozhatunk meg:

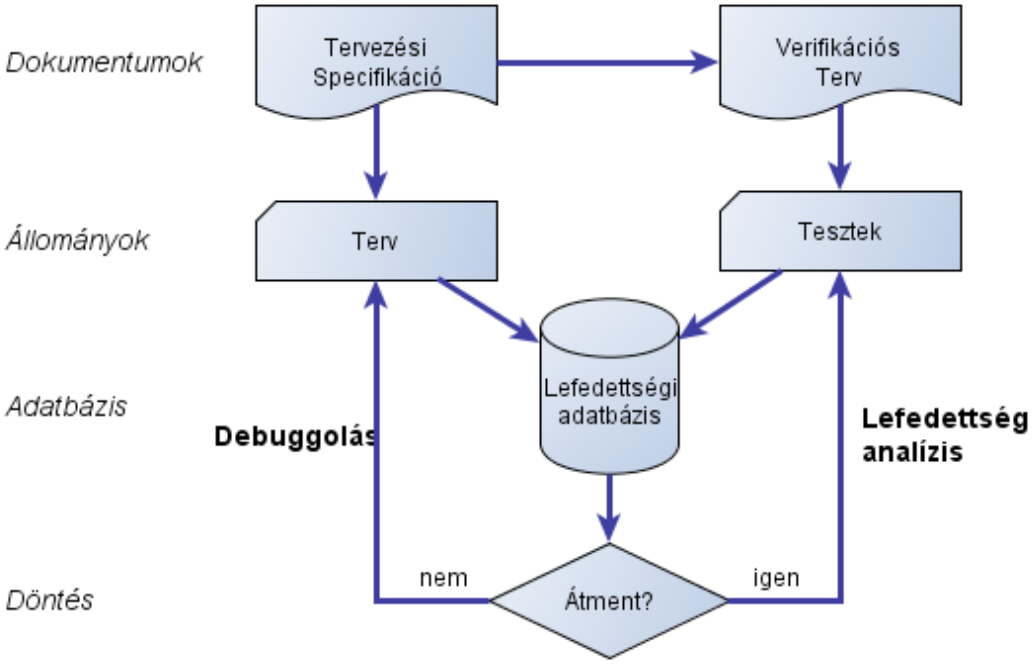
1.) Mely teszteseteket futtattuk eddig,
2.) Mely tesztesetek futottak le helyesen,
3.) A megtalált bugok száma és a megbízható attribútumok száma egy adott pillanatban,
4.) A tesztesetek minimális száma, amely futtatása szükséges ahhoz, hogy a szoftver minőségéről megbizonyosodjunk.

A fent említett néhány eset csak példa arra, mennyi mérőszámmal is bizonyítható egy szoftver megbízhatósága vagy éppen annak az ellenkezője. A mérések a következő módon végezhetők el:

1.) **Funkció-lefedettség**: a specifikációban rögzítésre kerül n darab szükséges funkció, a kérdés az, hogy minden funkcióra végre kell-e hajtani a tesztelést.
2.) **Grafikus felület lefedettsége**: a grafikus felület számos képernyőt, gombot, legördülő menüt, tabulátort, menüpontot stb. tartalmazhat. Szükséges nekünk az összeshez tesztesetet készíteni, valamint ezek közül mindet lefuttatni?
3.) **Instrumentálási lefedettség**: instrumentálási eszközöket buildek összehangolására használjuk, amelyeket később tesztelni tudunk. Az eszköz kimenetének képesnek kell lennie arra, hogy megmutassa, mekkora volt a lefedettség, és hogy ez elérte-e a kívánt szintet.
4.) **Szerkezeti lefedettség:** általában a unit tesztelést értjük alatta, vagyis a komponens szintű lefedettség tesztelését. A tesztelésnek magába kell foglalnia az utasításszintű, a döntési, a feltétel és a többszörös feltétel lefedettséget, valamint a Linear Code Sequence and Jump (LCSAJ) lefedettséget.
5.) **Forgatókönyv szerinti lefedettség**: a felhasználók számos célt kitűzhetnek maguknak a tesztelés céljaként, amit a funkciók számának felhasználásával érnek el. A forgatókönyv végrehajtásával számos olyan kisebb alfunkciót érinthetnek, amelyeket más lefedettségi mutatók nem tartalmaznak. A forgatókönyv szintű lefedettség felhasználói logokkal és profilokkal mérhető, rögzíthető. A lefedettség alapját a use case-ek, vagyis a felhasználói esetek képezik.
6.) **Tranzakció szintű lefedettség**: leggyakrabban webalkalmazásoknál használják, de használható hagyományosabb szoftvereknél is, ahol egy felhasználó egy adott viselkedést, célt számos útvonalon elérhet. Ezek a lehetséges útvonalak állapot-átmenet diagramon ábrázolhatóak, így könnyebben átlátható, hogy mely útvonalakon lehet áthaladni az egyes műveletek révén. A tranzakció szintű lefedettség tekinthető a struktúra és forgatókönyv szintű lefedettség keverékeként is.
7.) **Lefedettség webszkript, weboldal, alkalmazás és komponens szempontjából**. Amint beazonosítottuk a tesztelendő weboldal kockázati szintjét, kiválaszthatjuk, hogy milyen szinten mérjük a lefedettséget ahhoz, hogy a kockázati tényezőket mérsékeljük.

A lefedettség mérése, mint minőség vizsgálati tevékenység a fejlesztési környezetben, beleértve a szisztematikus, dinamikus tesztelést is, ritkán lehet kimerítő, ezért modellekkel reprezentáljuk őket. Például a statikus analízis attól a modelltől függ, melyet az elemzéshez definiáltunk. A partíciótesztelés attól a modelltől függ, melyet az egyes tesztelendő komponensek osztályozására hoztunk létre. Ezen modellek és a program aktuális viselkedése közötti eltérések értékes információként szolgálnak – még ha nem is vezetnek konkrét problémákhoz -, mivel megmutatják, hogy a fejlesztési környezetben végrehajtott minőségbiztosítási tevékenységek hogyan fejleszthetik a minőséget. Például, ha van valamilyen meghatározott ítélet, hogy mikor kell befejezni a tesztelést, vehető negatív értelemben is. A teszt elfogadási kritérium nem akkor jelez, mikor tesztelés mindenképpen megfelelő, hanem akkor, mikor bizonyíthatóan tesztek egy halmaza nem megfelelő, mert néhány fontos osztályhoz tartozó viselkedés nem került ellenőrzésre.

6 http://quality-gate.com/




*26. ábra Lefedettségi folyamat*

A strukturált lefedettségi kritériumok, mint pl. az utasításszintű lefedettség, döntési lefedettség, adatfolyam lefedettség a programvezérlés és adatfolyam szintaktikus modelljein alapszik. Mivel a szintaktikus modellek abban az értelemben konzervatívak, hogy nemcsak a futtatás során előforduló vezérlési és adatfolyamokat írnak le, hanem olyan végrehajthatatlan ágakat is, melyek soha nem fognak előfordulni. Éppen ezért a kimerítő tesztelés legtöbbször elbukik, mert a strukturális lefedettség kilépési feltétele nem teljesül.

A modelleket és feltételeket nem tudjuk teljes mértékben figyelmen kívül hagyni, amit tehetünk, az a használt modell validálása. Míg a fejlesztők számos monitorozó eszközt elérnek a fejlesztési környezetben, a monitorozás tipikusan a hiba- és alapvető ellenőrzésekre korlátozódik, és az egyetlen kommunikációs eszköz a felhasználók és a fejlesztők között a problémák listája. A maradék tesztlefedettség monitorozás kihasználja az egyre inkább széles körben elterjedő hálózatozás által nyújtott előnyöket, hogy gazdagítsa a kommunikációs csatornát és validáljon egy, a fejlesztési környezetben használt modellt.

A tesztlefedettség monitorozásának célja egy gazdagabb visszacsatolás biztosítása az alkalmazott szoftver aktuális használatáról a fejlesztők számára, ebben segítve őket a használt modell validásálásában és finomításában. A siker kulcsa a valós idejű monitorozás, amelynek meg kell felelnie a felhasználói igényeknek.

A kielégítendő igények egyik osztálya a biztonsághoz, bizalomhoz, és titoktartáshoz kapcsolódik, míg a másik problémacsoport a teljesítményhez, többek között a valós idejű viselkedés érzékenységéhez és zavarához kapcsolódik.

*Forrás: http://ix.cs.uoregon.edu/~michal/pubs/ResTestICSE99.pdf*

***Fogalmak a 7.4. fejezetben:*** utasításszintű lefedettség, döntési lefedettség, feltétel lefedettség, többszörös feltétel lefedettség, Lindear Code Sequence, LCSAJ, statikus analízis, partíciótesztelés, adatfolyam lefedettség, validálás.

# 8. BESZÁLLÍTÓI KÉRDÉSEK

## 8.1 Átadás-átvétel

Előfordulhat, hogy a megrendelt vagy megvásárolt szoftverátadás után nem megfelelően működik. Ez nemcsak a felhasználók számára, hanem a szoftvert gyártó cég számára is igen kellemetlen. Ennek elkerülése végett formális elfogadási tesztre van szükség az éles környezetben. Fontos megemlíteni, hogy egy szoftvert nem lehet élesben telepíteni vagy kiadni, amíg a biztonsági kockázatot nem csökkentették egy előre meghatározott küszöb alá.

A szoftvert verifikálni és validálni érdemes azelőtt, mielőtt azt átadjuk a végfelhasználóknak. A verifikáció azt vizsgálja, hogy a rendszer tervezése helyesen történt-e, tehát hogy a megfelelő követelményeket állítottuk-e fel. Ezzel szemben a validáció azt vizsgálja, hogy jó rendszer készítettünk-e el, vagyis az megfelel-e a követelményeknek. Mivel az átadás-átvétel fázisában már nemigen változtathatunk a követelményeken, a validációt fontos kiemelnünk.

A validáció nemcsak üzleti szempontból, hanem biztonsági szempontból is fontos.

További nehézséget okozhat az integráció az éles környezet többi elemével, így azt is vizsgálnunk kell az átadás előtt, hogy az adott szoftver hogyan reagál más szoftverekkel, környezettel.

Bizonyos típusú teszteket (beleértve biztonsági tesztelést) érdemes független harmadik félre bízni, aki kívülről látja át a rendszert.

A formális elfogadási tesztet szabályozni kell, az alkalmazott folyamatok és minőségbiztosítási eljárások függvényében. Az elfogadási kritériumokat előre meg kell határozni, még tervezési fázisban. ISO 25000 szabvány segítségül hívható, de a legfontosabb kategóriák: Funkcionalitás, Performancia, Minőség, Biztonság.

Meg kell azonban említeni, hogy attól hogy biztonságosnak tervezték a szoftvert, még nem jelenti azt, hogy biztonságos is, mivel a fejlesztési környezet egy zárt környezet. Ha élesben telepítjük és használjuk a szoftvert, akkor a környezet valószínűleg nem fog megegyezni a fejlesztési környezettel, így lehetnek olyan szoftverek, amik befolyásolják a szoftver biztonságát.

### A formális szoftverelfogadás előnyei

A formális szoftvertesztelési folyamat beépítése a kiadási folyamatba lehetővé teszi, hogy biztonságos szoftvereket adjunk ki a fejlesztés befejezésével. Ez a folyamat mint egy utolsó ellenőrzőpont szerepel a folyamat végén, ahol még fény derülhet biztonsági, illetve egyéb hiányosságok jelenlétére.

Ha a szoftver nem felel meg ezen a teszten, nem lehet kiadni.

Végfelhasználói elfogadási teszt céljai:

- lehető legjobban szolgálja ki a megrendelői igényeket,
- a rendszer minőségi mutatói megfeleljenek a megrendelőnek,
- csökkentse a módosítási igények számát a bevezetés után, ezáltal csökkentse a termék összköltségét, karbantarthatóság ellenőrzése.
- funkcionális követelményeknek megfelelő termék a felhasználók megítélése alapján legyen elfogadva, azaz a végső jóváhagyást a rendszer használói tegyék meg.

Elfogadási teszt előfeltételei:

- A rendszer bevezetésre kész állapotban van.
- Fejlesztés és a migráció sikeresen befejeződött.
- A rendszer elérhető az éles üzemnek megfelelő környezetben az éles beállításokkal.
- Unit és integrációs tesztek sikeresek.
- Manuális és automatizált funkcionális tesztek sikeresek.
- Az éles rendszer felhasználóit reprezentáló felhasználói csoport kijelölésre került

Tesztelési feladatok:

- Üzleti igények felmérése.
- Elfogadási kritériumok megfogalmazása.
- Teszt terv elkészítése.
- Tesztesetek elkészítése.
- Tesztek futtatása.
- Eredmények rögzítése, kiértékelése.
- Üzleti célok elérésének ellenőrzése.

Az átadott eredménytermék minél szélesebb körben megvizsgálandó, pl. kódelemzés, tesztelés kódlefedettségének mérése. A folyamatos és online kódlefedettség-mérés szintén sokat segít a végtermék minőségének ellenőrzésében.

### 8.1.1 Elfogadás biztonsági szempontból

Ahhoz, hogy egy szoftvert biztonsági szempontból is elfogadjuk, a következő feltételeknek kell teljesülnie:

- a rendszer legyen biztonságos mind tervezés, mind a telepítés szempontjából,
- a meglévő, alapszintű védelmet mélységi védelemmel egészítse ki,
- a legkisebb jogosultsággal fusson,
- nem lehet meghamisítani vagy a műveleteket visszafordítani,
- az adminisztratív funkciókat és a biztonsági kezelés interfészei elkülönülnek,
- nem műszaki védelmet is biztosít.

Ezen felül néhány fontosabb elemet meg kell vizsgálni, mielőtt a belső munkatársak elfogadják a szoftvert, ezeket a következő ábrán láthatjuk.


*27. ábra Biztonsági szempontok*

**Teljesülési feltétel**: a funkcionális és biztonsági követelményeket már a szoftverfejlesztési életciklus elején össze kell gyűjtenünk, és már ebben a fázisban befejezettként kell hogy validáljuk és verifikáljuk őket. Ezen a szinten bizonyos mérföldköveket kell meghatároznunk, biztonsági szempontból nézzünk erre néhány példát:

- a fenyegetettségi modell elkészülése a tervezési fázis alatt,
- a követelmény nyomonkövetési mátrixának legenerálása, amely mind a funkcionális, mind a biztonsági követelményeket tartalmazza,
- a biztonsági tesztelés befejezése az alkalmazás tesztelés fázisában.

Az egyes mérföldkövek tartalmazzák az aktuális munkatermékeket is, mint pl. az RTM, a Fenyegetettségi modell, Kód felülvizsgálati riport, Biztonsági szerkezet-terv, Biztonsági teszt riport stb.

A változásmenedzsment a konfigurációs menedzsment egyik alhalmaza. A fejlesztési környezet változása, valamint a biztonsági architektúra átszervezése új támadásoknak adhat teret, növelve ezzel a kockázatot. Éppen ezért szükség van változásmenedzsmentre.

**Telepítés, piacra bocsátás engedélyezése**: a szoftver telepítése előtt kockázatelemzés elvégzésére van szükség, valamint a fennmaradó kockázatok meghatározására. A szoftver elfogadása során validálni kell azt, hogy maga a szoftver elfogadása nemcsak egy pipa egy listán, hanem felülvizsgálatok és felügyeleti tevékenységek során lett bizonyítva, hogy a rendszer valóban megfelelt a követelményeknek.

**Kockázatok elfogadása és kivételkezelési politika**: annak az esélye, hogy nem marad kockázat az elkészült szoftverben sajnos igen utópisztikus, így a megmaradó kockázatokat először meg kell határoznunk. A legjobb opció

a teljes kockázat megcímzése a megmaradó kockázat megbecsülése végett, így megtudhatjuk, hogy a megmaradó kockázat az előre definiált küszöb alá esik-e.

| **Kockázat elfogadási dokumentum #1** | KED_001 [egy egyedi azonosító] |
|---|---|
| **Kockázat osztályozása** | Magas, közepes, alacsony (meghatározott szintek, tetszőleges mértékekből állhat) |

**Kockázat**: [Itt írjuk le részletesen a kockázatot, lehetőleg technikai kifejezések nélkül annak érdekében, hogy az üzletemberek is megértsék]

**Cselekvés**: [Ebben az részben a szükséges lépéseket írjuk le]

**Tényezők**: [Itt a fenyegetések részleteit fejtük ki szakszavak és technikai szöveg használatával, valamint azt, hogy ezek hogyan nyilvánultak meg]

**Döntések**: [Itt a kockázat kezelésére használható lehetséges döntési alternatívákat és véleményeket fejtjük ki, beleértve a biztonsági intézkedéseket]

**Hitelesítés**

| | **Döntés** | **Indoklás** |
|---|---|---|
| X | Elfogadás | |
| | Áthelyezés | |
| X | Becslés | |
| | Figyelmen kívül hagyás | |

Dátum: .....

---------- ----------

Kovács János
projektvezető

Teszt Elek
tesztmenedzser

*28. ábra Kockázatelfogadási sablonra példa*

**Dokumentáció**: Fontos, hogy az egyes fejlesztési fázisok végén is elkészüljön a dokumentáció.

***Fogalmak a 8.1. fejezetben:*** elfogadási teszt, biztonsági kockázat, verifikálás, validálás, kódlefedettség, változásmenedzsment.

## 8.2 Szoftverevolúció

A legfontosabb tudnivaló egy szoftver javításával, módosításával kapcsolatban, hogy egy szoftver soha nincs készen, hiszen a felhasználói, technikai igényeknek megfelelően folyamatos változtatása lehet szükséges. Minden változtatás a szoftveren újabb kockázatokat rejt, ezért megfelelő folyamatok mentén kell azt végezni.

### 8.2.1 Telepítés és terítés

Mikor a nem megfelelő telepítési és terítési folyamatokat követjük, nagy esély van arra, hogy a rendszer vagy annak környezete kevésbé lesz biztonságos. Az elő- és utó-telepítés biztonsági megfontolások nélkül nem eredményezhet egy támadásokat rugalmasan kezelő rendszert.

A szoftver konfigurálására szükség van, így a biztonsági szabályok, mint pl. a legkisebb jogosultság, a mélységi védekezés, a feladatok elkülönítése stb. is figyelembe vételre kerülnek a telepítési fázisban.

A legfontosabb elő- és utótelepítési tevékenységek a következők lehetnek:

- „Megkeményítés" (Hardening)
- Környezeti konfiguráció (Environment Configuration)
- Értékesítés Menedzsment (Release Management)
- Rendszertöltés és biztonságos üzembe helyezés (Bootstrapping and Secure Startup)

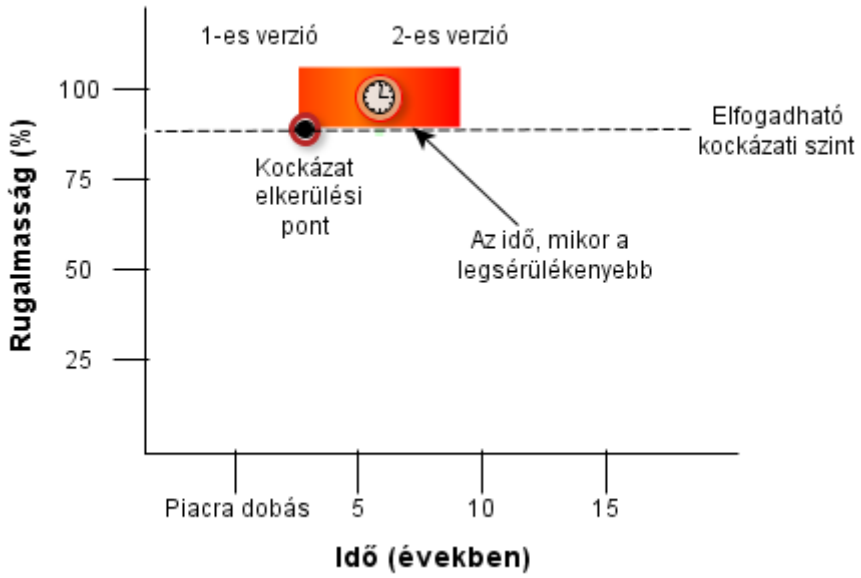
**Értékesítés management**

Amint a hardver és a szoftver erőforrások keresztül jutottak a megerősítésen, és a környezet biztonsági szempontból történő konfigurálásán, a szoftvert egy üzemi számítási környezetbe kell helyezni. Az értékesítési menedzsment az a folyamat, amelynek során megbizonyosodunk arról, hogy minden változás, amely érinti az üzemi-számítási környezetet, túljutott a tervezésen, dokumentáción, a tesztelésen és a legkisebb jogosultsággal történő telepítésen, negatív hatást nem gyakorolva a létező üzleti folyamatokra, felhasználókra, vásárlókra és a felhasználó, támogató csapatra.

Az értékesítés menedzsmentben a leggyakoribb hiba, hogy azok a bugok, amelyek rögzítve lettek, újra felbukkannak. Ennek elsődleges oka a nem megfelelő verziókövetés. Szintén fontos, hogy a szoftver értékesítésre készítsük, és ne debuggolásra. Mikor egy szoftvert debuggolási célból állítunk össze, a debuggolási információkat általában egy különálló fájlban tároljuk, amely a program adatbázis-fájljaként ismeretes. Ezt a fájlt nem ajánlott a kibocsátott szoftvercsomagban hagyni, ugyanis hibakeresési és fontos projekt információkat tartalmazhat, amelyeket a rosszindulatú felhasználók a rendszer károsítására használhatnak fel.

### 8.2.2 Az üzemeltetés biztonsági megfontolásai

Amikor a rendszert telepítettük, a rendszer üzemelése során szolgáltatásokat biztosít az üzlet vagy a végfelhasználók számára. A szoftver üzemeltetése és karbantartása szempontjából számításba kell venni a megbízható, rugalmas és hasznos adatfeldolgozás aspektusait. Amíg a teljes biztonság zéró kockázattal egy idilli állapot, addig minden alkalmazott szoftver tartalmaz hátramaradó kockázatokat, melyek gyakran az elfogadható szint alatt vannak. A leghatékonyabb erőfeszítések ellenére is a telepített szoftver még mindig tartalmazhat ismeretlen támadási pontokat és adatbiztonságot kockáztató tényezőket. A szoftver rugalmasságának ezért mindig az előre meghatározott kockázati küszöb felett kell lennie.




*29. ábra A rugalmassági szintek az idő múlásával*

Amikor az üzemeltetés a kockázat becslése nélkül folytatódik, majd a következő verzió kerül kifejlesztésre az az időszak, amikor a rendszer a legsebezhetőbb.

Az erőforrásokat több csoportba oszthatjuk, mint pl. hardver, szoftver, média és emberi erőforrások. Hardver erőforrásokra példák:

- hálózati eszközök, mint pl. switch-ek, routerek stb.
- kommunikációs eszközök, pl. telefon, fax, VoIP eszközök stb.
- számítási eszközök, pl. szerver, munkaállomás, PC, laptop stb.

Szoftver erőforrások a következők lehetnek:

- belső fejlesztésű szoftverek,
- külső előállítású szoftverek,
- adatok,
- operációs rendszerek.

## 8.2.3 Backup, recovery, archiválás

Az üzlet folyamatossága megzavarások nélkül egy fontos tényezője a biztonságos üzemeltetésnek. A nem megfelelő, valamint nem biztonságos műveletek elvégzése az adatokat és magát a rendszert is elérhetetlenné teheti a jogosult felhasználók és folyamatok számára. Létezik néhány tevékenység arra, hogy a műveltek elvégzését biztonságossá tegye, ezek a biztonsági mentések készítése, helyreállítás és az archiválás.

Ráadásul a rendszeresen ütemezett biztonsági mentések mellett, mikor szoftverfrissítésekre kerül sor, érdemes az egész rendszerről egy biztonsági mentést készíteni, majd verifikálni azt. Ez azért fontos, mert ha a frissítések nem várt következményeket okoznak, vissza lehet állítani a rendszert, így a váratlan események csak minimális káros hatást gyakorolnak a működésre.

A helyreállítási folyamatokat szintén kontrollálni kell. Csak a nagyobb jogosultsággal rendelkezők számára érdemes engedélyezni a visszaállítást és a mentések betöltését. Ez különösen fontos abban az esetben, mikor a visszaállítandó adatok magánjellegűek vagy igen érzékeny természetűek.

Az archívumok jól jöhetnek a felhasználói támogatásban, különösen a korábbi vásárlók tekintetében.

## 8.2.4 Karbantartás fajtái

A karbantartást többféleképpen lehet elvégezni. Lehetséges fajtái:
- Adaptív karbantartás: ahol a fejlesztések szükségesek ahhoz, hogy a működő eljárásokat megváltoztassuk.
- Javító karbantartás: ahol folyamatos javítások során bizonyosodunk meg arról, hogy a rendszer továbbra is megfelel az eredeti követelményeknek.
- Perfektív karbantartás: a rendszer fejlesztése annak érdekében, hogy az sokkal kifinomultabb és hatékonyabb legyen.
- Megelőző karbantartás: stratégiai változtatások eszközlése a várakozásoknak megfelelően.

## 8.2.5 Evolúciós folyamat

A szoftver evolúciójának lépései a következők:
1.) Változtatási igény
2.) Hatásvizsgálat
3.) Kiadás megtervezése
4.) Változás megvalósítása
   *a)* követelmények analízise
   *b)* követelmények frissítése
   *c)* szoftverfejlesztés
5.) Kiadás

**1.** A **változtatási igényeknek** több célja is lehet, ilyen pl. a hibajavítás. A hibajavítás során sor kerül auditálásra, a hiba jelentésére, melyet tesztelési tevékenységgel vihetünk véghez. Szintén cél lehet a belső minőségjavítás, más néven refaktoring. A javítandó kódrészeket bad smell-ek alapján ismerhetjük fel, majd azt javíthatjuk a kód átstrukturálásával. Vigyázzunk arra, hogy az átrendezések-módosítások során a külső viselkedés ne változzon! Harmadik oka a változtatási igényeknek a továbbfejlesztés, adaptáció lehet. Ezeket a felhasználók igényei szerint végzik el a fejlesztők. A változtatások helyét minél gyorsabban be kell azonosítani, hogy a módosításokat minél hamarabb el tudják végezi a megfelelő fejlesztők.

**2.** A **hatásanalízis** lényege annak vizsgálata, hogy egy modul megváltoztatása hogyan érint más modulokat, valamint annak környezetét. Teljesen biztonságos módszert még nem dolgoztak ki, ugyanis szemantikus összefüggések is lehetnek az egyes komponensek között. Viszont már vannak olyan közelítő módszerek, amelyek hatékonyan támogathatják a hatásanalízist.

A hatásanalízis módszerei lehetnek a következők:
- *Forráskód (vagy modell) elemzésén alapuló módszerek*: fő módszere a függőség analízis, amely történhet statikusan vagy dinamikusan. A szemantikus, rejtett összefüggések felfedezésére sajnos nem alkalmas.
- *Nyomonkövethetőségi információn alapuló módszerek*: lényegük, hogy különböző programreprezentációk és kapcsolódó dokumentációk közötti kapcsolatokat vizsgálnak, pl. a követelmények és kód között.
- *Adatbányászat egyéb forrásból*: más forrásokból is szerezhetünk információkat, pl. konfigurációs menedzsment eszközökből.

**3.** Harmadik lépés a **kiadás megtervezése**, mely a szoftvermenedzsment feladata.

**4.** A megtervezett **változtatásokat el kell végezni**, ennek módja a változás minőségétől függ. Előfordulhat, hogy a modellt kell módosítani, vagy kódolást kell végezni, esetleg refaktoring végrehajtásával a kód minőségét javítani. Fontos, hogy egy modul megváltoztatása után az érintett további modulokat is hozzá kell igazítani, ezt hatásanalízis alapján végezhetjük el. A cél a minél teljesebb propagálás.

**5**. A **kiadás** során a változtatásokkal bővült termék kerül kibocsátásra.

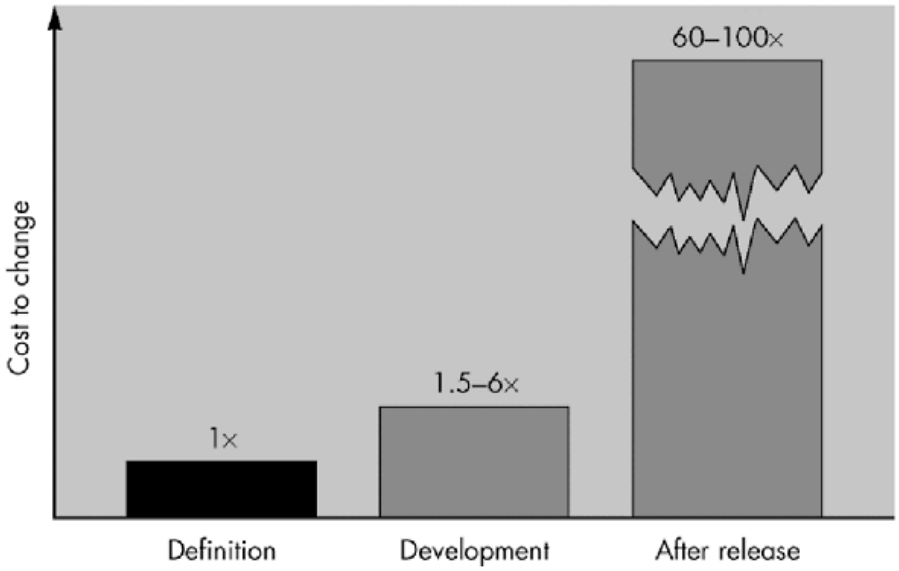
## 8.2.6 Változtatás management

A probléma gyökerének meghatározása után meg kell határozni a visszaállítás miként jét és a probléma megoldását. A változtatási igény ezután inicializálásra kerül. Ezen felül fel kell ismerni, hogy a problémamenedzsment gyakran eredményez változásokat a belső folyamatokban, procedúrákban vagy az infrastruktúrában. A Build Security In

Maturity Model-re hivatkozva, a teljes célja a konfigurációnak és sérülékenység-menedzsmentnek a változásmenedzsment.

Mikor változást szükséges véghezvinni a problémamenedzsment tevékenységei során, érdemes követni a változásmenedzsment folyamatait és protokolljait a vállalat által meghatározottak szerint. A megfelelő változtatási kérelmet a fő probléma (root cause) azonosítása után kell elvégezni, még abban az esetben is, ha a probléma megoldása nem tartós.

Egy szoftvert a kibocsátás után sokkal nehezebb és költségesebb megváltoztatni, viszont kibocsátás után is fedezhetünk fel hibákat, valamint a felhasználók bombázhatják a fejlesztőket további kérésekkel.




*30. ábra Változások költségei az egyes fejlesztési szakaszokban*

Ebben az esetben nem szabad a teljes rendszert eldobni, hanem bizonyos frissítő vagy javító csomagokkal javítható annak minősége.

Csak az arra jogosultak által kezdeményezett változtatásokat szabad engedélyezni, és a csaló, jogosultatlan módosításokat kimutatni és kezelni kell. A változtatások implementációját követve fontos, hogy nyomon kövessük a biztonsági réseket és monitorozzuk a probléma megoldását annak érdekében, hogy megbizonyosodjunk annak hatékonyságáról és arról, hogy a probléma többé nem fordul elő.

## ISO 25000

Az ISO 25000:2005 szabvány útmutatást nyújt a Standards named Software product Quality Requirements and Evaluation (SQuaRE) használatához. A dokumentum célja egy általános rálátás biztosítása a SQuaRE tartalmára, közös referencia modellekre és definíciókra, valamint az azok közötti kapcsolatokra.

## Termékminőség metrikák

A termékmetrikák dinamikusak abból a szempontból, hogy szoros kapcsolatot ápolnak az egyes minőségi jellemzőkkel, mint pl. a teljesítménnyel, hibák számával. Statikusak viszont abban, hogy közvetett kapcsolatban állnak a kóddal.

A termékmetrikáknak több típusa van: lehetnek nyelv-függetlenek vagy nyelv-specifikusak. A típusuk szerint megkülönböztetünk többféle metrikát is:

- méret metrikák (LOC, NA, NM): a rendszer, valamint a rendszer elemeinek méretét vizsgálják.
- öröklődés metrikák (DIT)
- komplexitás metrikák (McCabe, WMC)
- kohéziós metrikák (LCOM)
- csatolás metrikák (CBO)
- bad smell és copy-paste metrikák (CC)
- kódolási minőség metrikák (szabálysértések)

**Bad smellek**

A szoftver fejlesztése során keletkezhetnek olyan részek a kódban, amelyek kevésbé hatékonyak, valamint sebezhető részei a rendszernek. Ezeket „bad smell”-eknek nevezzük. A bad smell nem egy metrikus érték, hanem a kód azon részeinek megjelölésére szolgál, amelyek valamilyen szempontból problematikusak. Ezek a részek nem mindig okoznak, valamint nem mindig jelentenek tényleges hibát, pusztán felhívják a figyelmet olyan pontokra a kódban, melyeket érdemes felülvizsgálni. Az egyes bad smellek javítása komoly refactoringot igényelhet. Példák bad smellre:

- Data Class – adatosztály
- Large Class – nagy osztály
- Lazy Class – lusta osztály
- Long method – hosszú eljárás
- Temporary Field – ideiglenes mező

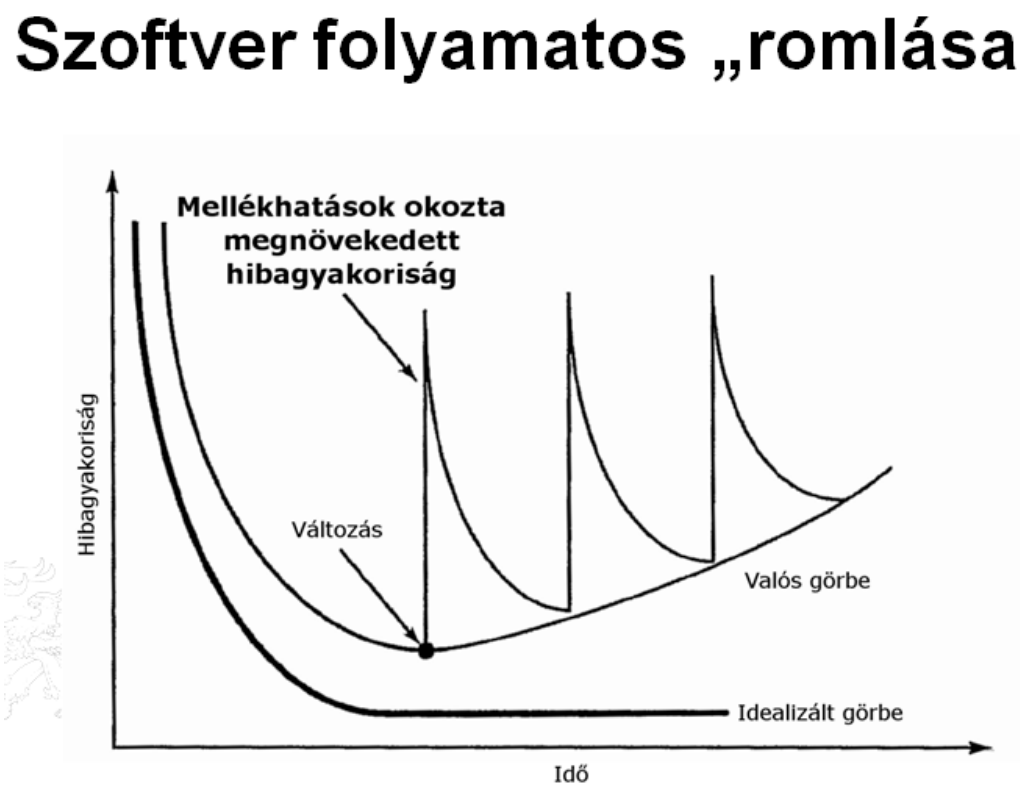
**Refactoring**

A refaktoring egy szabályozott technika létező kód átstrukturálására, megváltoztatva annak belső szerkezetét anélkül, hogy a külső viselkedés módosulna.

A refactoring sok viselkedést megőrző, kis változtatási lépés sorozata. Minden egyes transzformáció (más néven refaktoring) egy apró módosítást végez csak el a kódon, viszont több transzformáció használata jelentős változást hajthat végre azon. Mivel a változtatásokat csak apránként hajtjuk végre, kisebb az esélye annak, hogy a módosítások súlyos következményekkel járnak. A módosítások után a rendszer szerkezete „letisztul”, kisebb hibalehetőséget tartalmazva, a rendszer pedig megtartja eddigi viselkedését.

## 8.2.7 Életciklus vége, selejtezés

A szoftver képessége a támadások ellenállására az idő múlásával csökken még akkor is, ha újabb támadásokat és fenyegetéseket fedezünk fel, vagy ha technikai haladásban történő változásokat eszközölve nagyobb biztonsági védelmet tudunk nyújtani.




*31. ábra A szoftver romlása az idő múlásának függvényében*

Addig, amíg a szoftver üzemel, mindig maradnak fent kockázatok, amelyeket kezelni kell egészen a szoftver kivezetéséig. A szoftver kiiktatására gyakran hivatkoznak visszavonásként, annak alkonyataként vagy leszereléseként.

Az első követelmény a szoftver kiiktatásához egy End-of-Life (EOL) dokumentum megléte, amely részletesen leírja a szoftverek eltávolításának vagy áthelyezésének menetét, ügyelve a kezelt adatok biztonságos eltávolítására vagy további tárolására. Az EOL stratégiának tartalmaznia kell a

- kivezetési kritériumot,
- minden hardver és szoftverelemet, amely kivezetésre vagy áthelyezésre kerül,
- a technikai támogatás időtartamának ismertetését, hogy a felhasználók tudják, hogy még mennyi ideig tudják használni a kivezetendő szoftvert és annak adatait.

***Fogalmak a 8.2. fejezetben:*** értékesítés menedzsment, backup, recovery, archiválás, hatásanalízis, változásmenedzsment, bad smell, refactoring.

# 8.3 Monitoring rendszerek

A motiváció: ha nem monitorozunk, nem tudunk mérni, felügyelni és így nem tudjuk menedzselni a munkafolyamatokat. A biztonsági menedzsment tevékenységek részeként a folyamatos monitorozás kritikus fontosságú.

## 8.3.1 Monitorozás előnyei

Mit monitorozzunk? Monitorozhatunk teljes rendszereket, szoftvereket vagy csak egyes folyamatokat. A monitorozás előnyeit számos helyzetben kamatoztathatjuk, mint pl.

- a követelményeknek és egyéb szabványoknak való megfelelés validálása,
- bizonyítékok mutatása auditálási védelemre,
- annak validálása, hogy a megfelelő ellenőrzési mechanizmusok a helyükön vannak és hatékonyan működnek,
- külső és belső támadások detektálása,
- a teljes biztonsági helyzet validálása stb.

## 8.3.2 Monitorozási lehetőségek

A monitorozás elsődleges módjai napjainkban a következők:

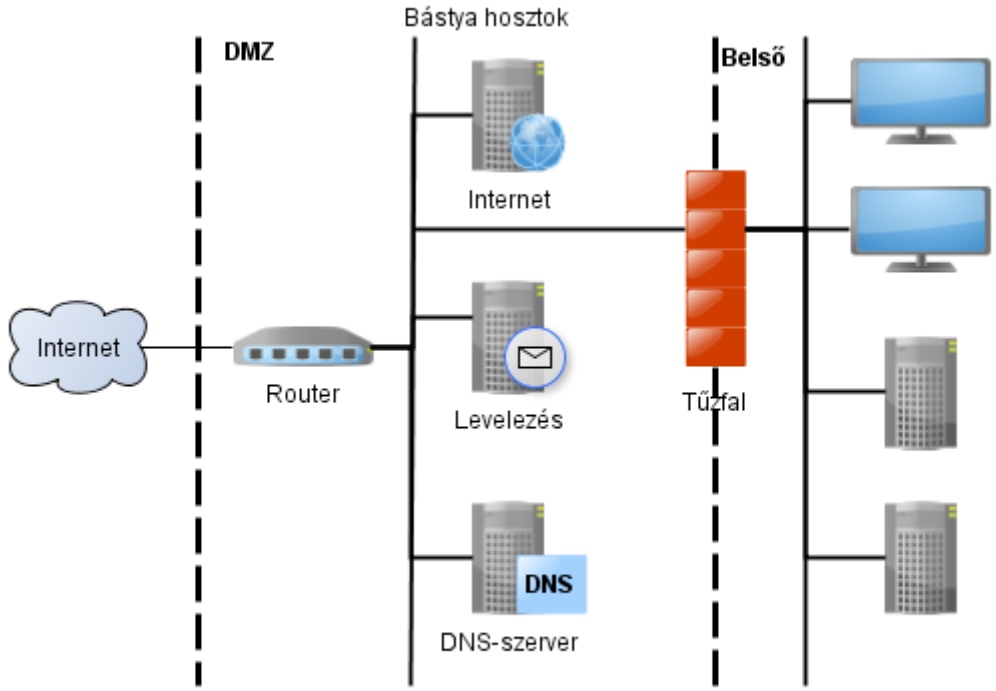
- letapogatás (scanning),
- naplózás (logging),
- behatolás jelzés (intrustion detection).

A **letapogatás** célja a számítási ökoszisztéma alkotóelemeinek meghatározása és újabb fenyegetések detektálása a környezetben.

A felhasználói tevékenységek **naplózása** kritikus az adat-kiegyezés hatásainak megelőzésében, detektálásában és enyhítésében. Általában öt fontosabb biztonsági célt szoktak megfogalmazni, amelyeket naplózással és nyomon követéssel lehet megvalósítani:

- legyen lehetőség az elérési sablonok és múltbéli események, valamint a különböző védelmi mechanizmusok jelenlétének és hatékonyágának felülvizsgálatára,
- legyen lehetőség belső és külső fenyegetettségek felfedezésére, amelyek megpróbálják megkerülni a biztonsági ellenőrző rendszert a rendszerben vagy a szoftverben,
- legyen lehetőség a legalacsonyabb jogosultsággal történő szabálysértések felfedezésére,
- legyen lehetőség elrettentő hatást gyakorolni a potenciális veszélyforrásokkal szemben,
- legyen lehetőség a biztonsági stratégia megsértéseinek enyhítésére.

**Behatolás jelző rendszerek**et olyan potenciális támadások és fenyegetések monitorozására használják, amelyek a tesztelendő rendszert veszik célba. A monitorozás részeként, gyakran bástyaként használják a valódi fenyegetések ellen, melyek a hálózatot használják közlekedési csatornának. A számítástechnikában a „bástyahoszt” egy olyan megerősített rendszer, amely teljes mértékben ki van téve a külső támadásoknak és illegális belépési kísérleteknek, ezzel védve az eredeti rendszert.




*32. ábra Bástya hosztok*

Amint az ábrán is látható a bástya hosztokat nem védi tűzfal. Ezen hosztok üzembe helyezésének tervezését nagy figyelemmel kell elvégezni, mivel a bizonytalan tervezés könnyen rosszindulatú, külső támadások áldozatává válhat. A bástya hosztokat meg kell erősíteni, és minden fölösleges szolgáltatástól, protokolltól, porttól, programtól mentesen kell azt üzembe helyezni.

### 8.3.3 Monitorozási metrikák

Fontos, hogy ezeket a metrikákat előre pontosan meghatározzuk. Egy szoftverre akkor mondhatjuk, hogy optimálisan és biztonságosan működik, ha az megfelel a metrikákban előre megfogalmazott optimális és biztonságos értékeknek.

A Service Level Agreement gyakran tartalmazza ezeket a mutatókat, de a mutatók nemcsak kizárólag az SLA-ban meghatározottak lehetnek.

A jó metrikák segítenek átfogó biztonsági döntéseket hozni, míg a rossz metrikák nem. A jó metrika tulajdonságai a következők:

**Következetesség**: azt jelenti, hogy nem számít, hányszor mérjük meg a metrikát, minden alkalommal ugyanazt az adathalmazt használva, ugyanazt az eredményt kell kapnunk. Tehát nem lehetnek jelentős eltérések az egyes mérések eredményeiben.

**Mérhetőség**: azt jelenti, hogy egy metrika precízen kerül kiszámolásra valamilyen szám vagy százalék formájában.

**Objektivitás**: azt jelenti, a metrikák összegyűjtéséért felelős személytől függetlenül az eredményeknek a valós helyzetet kell tükröznie. Vagyis a metrikák adjanak információkat, ne pedig az információkból adódjanak a metrikák.

**Kontextus-specifikus metrikák**: nemcsak azt teszik lehetővé, hogy használható és kellően informált döntéseket hozzunk, hanem hogy trend-információk is meg tudjunk határozni.

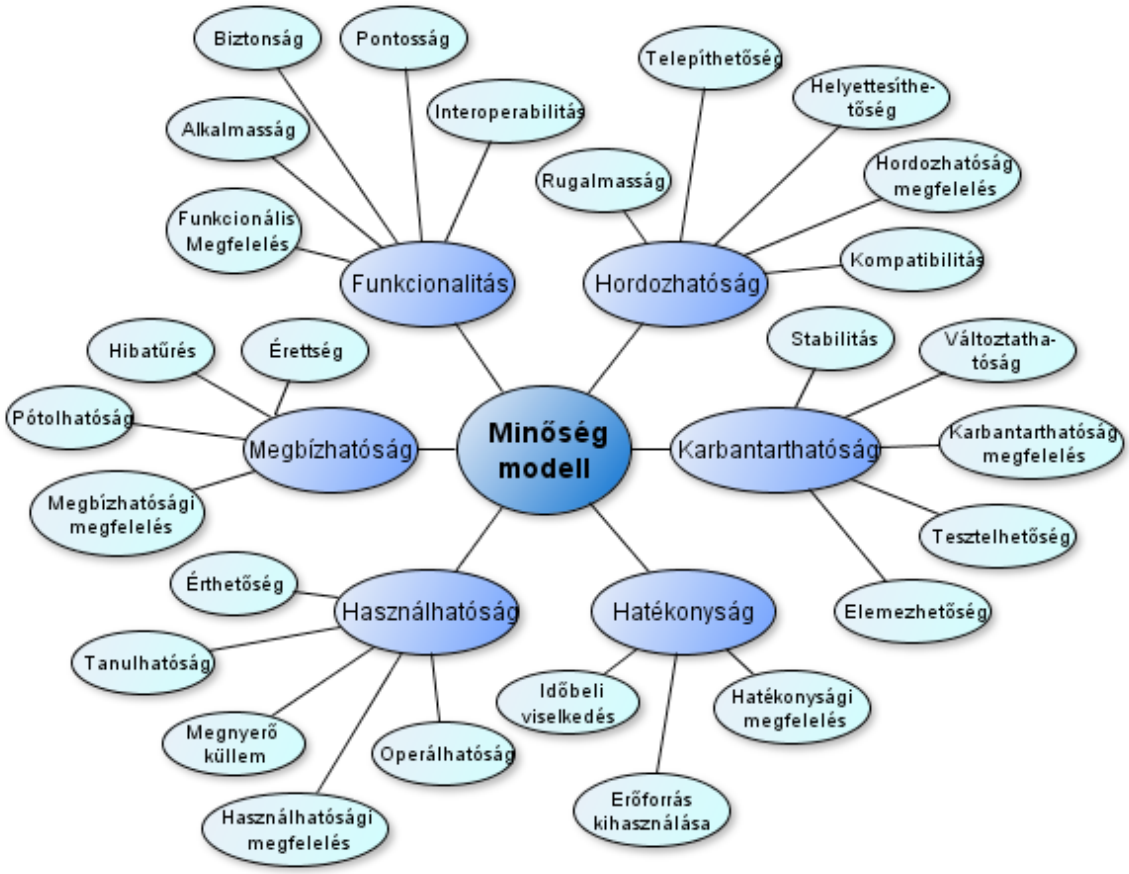
**Olcsó metrikák**: azt jelenti, hogy a metrikát gyakran automatizált eszközökkel gyűjtik össze, amely sokkal olcsóbb, mint az információk manuális gyűjtése.

| Tulajdonság | Jó metrikák | Rossz metrikák |
|---|---|---|
| *Gyűjtés* | Következetes | Nem következetes |
| *Kifejezés* | Mért (szám vagy %) | Minősített (értékelések) |
| *Eredmény* | Objektív | Szubjektív |
| *Relevancia* | Kontextus-specifikus | Kontextus irreveláns |
| *Ár* | Olcsó (automatizált) | Drága (manuális) |

## 8.3.4 Folyamat metrikák

A folyamat metrikák megmutatják az aktuális folyamat hatékonyságát, így a menedzsment láthatja, hogy mi az, ami jól megy és mi az, ami nem. A folyamat mutatókat hosszú távon minden projektre kell gyűjteni.

Az ISO 25000 elődje az ISO/IEC 9126 szabvány, amely a Szoftverfejlesztés – Termék minőség nevet viseli, amely egy nemzetközi szabvány a szoftver minőségének értékelésére. A szabvány által definiált minőségmodell több fő elemmel is rendelkezik, amelyek további alelemekre bonthatóak.




*33. ábra Minőség-modell*

Ezek közül biztonság szempontjából a legfontosabbak a megbízhatóság kritériumai – olyan jellemzők halmaza, amely a rendszer azon képességét méri, hogy a teljesítményt milyen szinten tudja karbantartani adott feltételek mellett egy meghatározott időintervallumon belül.

További példa lehet egy metrikára a rendelkezésreállási metrika, ami azt mutatja, hogy milyen arányban elérhető az adott rendszer a teljes üzemideje alatt. Egy másik metrika lehet, hogy a szoftver fejlesztése során a különböző verziókkal hogyan változott a biztonság, illetve a stabilitás.

Amint már fentebb is említettük, a jó metrikákhoz automatikusan érdemes gyűjteni az információkat, így időt és pénzt takaríthatunk meg. Az automatikus információmérésre számos eszköz létezik, melyek közül találhatunk mind ingyenes és nyílt forráskódú, mind kereskedelmi termékeket.

***Fogalmak a 8.3. fejezetben:*** monitorozás, letapogatás, naplózás, behatolás jelzés, bástyahoszt, Service Level Agreement (SLA).

# 8.4 Incidens Menedzsment

## 8.4.1 Az Incidens Menedzsment általánosságban

Az Incidens Menedzsment (röviden IcM) az IT Szolgáltatás Menedzsment (ITSM) egy folyamatterülete. Az elsődleges célja a normál szolgáltatás visszaállítása, amilyen gyorsan csak lehet, valamint az üzleti folyamatok káros hatásainak minimalizálása, így biztosítva azt, hogy a szolgáltatásminőség és elérhetőség a lehető legjobb szinten van karbantartva. Felismerési, elemzési tevékenységeket értünk alatta, valamint a lehetséges veszélyek megelőzését annak érdekében, hogy azok a jövőben ne forduljanak újra elő.

Az incidens egy olyan esemény, amely során a szoftver viselkedése eltér az elvárt működéstől, és ez további működési zavarokhoz vagy adatvesztéshez vezethet. Ha nem kezelünk le egy incidenst, az vészhelyzetet, krízist vagy akár katasztrófát is okozhat. Az Incidens Menedzsment az egyes események által okozott potenciális zavarok limitálásának folyamata abból a célból, hogy a szoftver működése megfelelő legyen.

Hatékony incidenskezelés nélkül egy ilyen esemény gyorsan megzavarhatja az üzleti folyamatokat, az információ biztonságot, az informatikai rendszereket, alkalmazottakat vagy ügyfeleket, valamint egyéb létfontosságú funkciókat.

Az Incidens Menedzsment főbb tevékenységei a következők:

- **Azonosítás**: az incidens keresése vagy bejelentése,
- **Regisztrálás**: az incidens regisztrálása,
- **Kategorizálás**: az incidens kategorizálása prioritás, SLA, vagy más tulajdonság alapján,
- **Priorizálás**: az erőforrások jobb kihasználás érdekében az incidenseket priorizálni kell,
- **Diagnózis**: az incidens összes lehetséges hatását mutatja meg,
- **Kiterjesztés**: akkor van rá szükség, ha szükség van több szervezeti egységre (legyen az munkatárs vagy valamilyen eszköz),
- **Vizsgálat és diagnózis**: ha nincs létező megoldás az incidens által okozott problémára, annak vizsgálatára van szükség a probléma gyökerének megtalálására,
- **Felbontás és helyreállítás**: amint megtaláltuk a megoldást, az incidens megoldottnak minősül,
- **Incidens lezárása**: az incidens státusza befejezettnek tekinthető.

## 8.4.2 Az Incidens Menedzsment biztonsági kontextusban

Amíg a folyamatos monitorozás lényege a betörési vagy károkozási kísérletek folyamatos figyelése, addig az Incidens Menedzsment a megfelelő protokollok követését célozza meg abból a célból, hogy a megfelelő lépéseket vigyen véghez a biztonsági rések vagy kockázat felbukkanásakor. Kezdve az incidens detektálásától, amely monitorozással is megvalósulhat, ehhez incidens detektáló és megelőző rendszereket vagy más mechanizmusokat használva az első lépés az incidens reagálására annak megállapítása, hogy az incidens valóban egy károkat okozható incidens vagy sem. Ha valid incidensről beszélünk, annak típusának meghatározására van szükség.

Ezután a veszteségek minimalizálására, és a gyengeséget javítására, elkerülésére, eltávolítására vagy orvoslására kell koncentrálnunk.

Fontos odafigyelnünk az incidensek rögzítésére és riportok készítésére.

## 8.4.3 Események, figyelmeztetések, incidensek

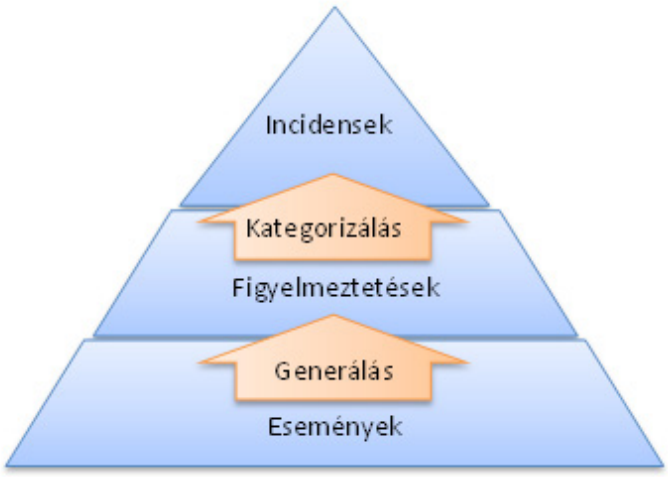
Az incidens felfedezése után definiálni kell azt, hogy mi okozta az incidenst. Az események és figyelmeztetések incidensként való téves besorolása komoly következményekkel és költséggel járhat. Így fontos meghatározni a következők közötti kapcsolatot:

– Események,
– Figyelmeztetések,
– Incidensek.

Minden olyan intézkedés, amely arra irányul, hogy megváltoztassa egy objektum állapotát, eseménynek nevezünk. Más szóval, egy esemény megfigyelhetően előfordulhat hálózatokban, rendszerekben vagy szoftverekben. Mikor egy eseményt észlelünk, további elemzésre van szükség ahhoz, hogy lássuk, az események megfelelnek bizonyos sémáknak vagy feltételeknek. Mikor egy esemény egyezést mutat az előre meghatározott feltételekkel vagy sablonokkal, figyelmeztetéseket (vagy red flag-eket) generálnak.

Azokat az eseményeket, melyek negatív vagy hátrányos következményekkel járhatnak, ellenséges eseményeknek hívjuk. Néhány példa ellenséges eseményekre a leterhelt hálózatok, rootkit telepítés, nem ellenőrzött belépési mechanizmus, kártékony kód vagy üzleti zavarok. A figyelmeztetések olyan megjelölt események, amelyeket tovább kell vizsgálni, hogy megállapítsuk, azok incidensek-e vagy sem.

A figyelmeztetéseket több kategóriába tudjuk sorolni: incidensek és ellenséges események. Ezen belül beszélhetünk biztonsági incidensekről is, amennyiben azok megszegik a hálózat vagy a rendszer biztonsági szabályait.




*34. ábra Kapcsolat az események, figyelmeztetések és incidensek között*

Az események, figyelmeztetések és incidensek kapcsolatát hierarchikusan tudjuk ábrázolni, ahol minden incidens egy figyelmeztetés, és minden figyelmeztetés egy esemény.

## 8.4.4 Incidensek típusai

Az incidenseknek számos típusa van, a fő biztonsági incidensek az alábbi típusúak lehetnek:

### Szolgáltatás megtagadása (Denial of Service, DoS)

Állítólag a legismertebb típusa a biztonsági incidenseknek. A DoS egy olyan támadás, amely egy jogosult felhasználót akadályoz vagy károsít a hálózaton, rendszerben vagy alkalmazásban az erőforrások leterhelésével.

### Kártékony kód

Az ilyen típusú incidensek a kódot károsítják, lehetnek pl. vírusok, férgek vagy trójai falovak, melyek sikeresen megfertőzik a hosztot.

### Jogtalan belépés

A beléptetéssel kapcsolatos incidensek, amelyek során egy személy logikai vagy fizikai belépési lehetőséghez jut hozzá rendszerekbe, hálózatokba, alkalmazásokba, vagy más IT forrásokba anélkül, hogy lennének jogai hozzá.

### Nem megfelelő használat

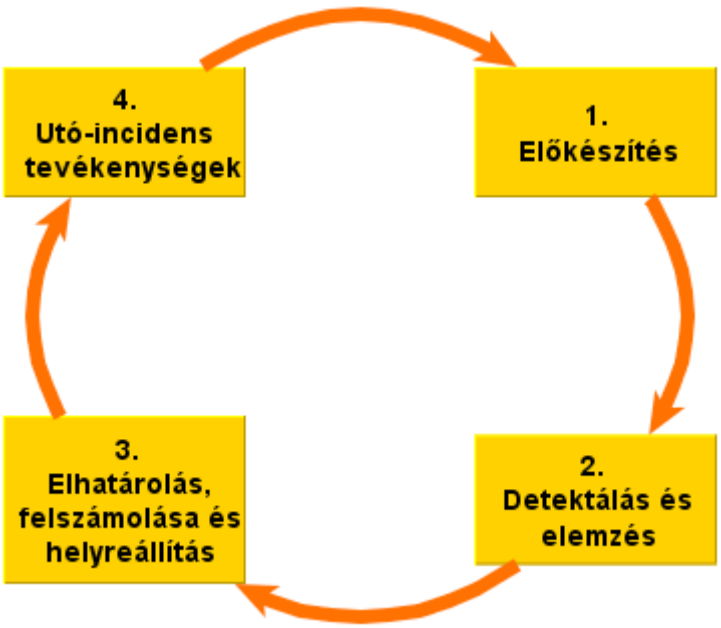
A nem megfelelő használathoz köthető incidensek azokat az eseteket tartalmazzák, mikor egy személy megszegi a rendszer elfogadható használatának szabályait.

### Többszörös tartalom

A többszörös tartalomra vonatkozó incidensek kettő vagy több incidenst is tartalmaznak. Vegyünk példának egy SQL injekciót, amely az alkalmazás réteget kihasználva engedélyezi a támadónak, hogy hozzáférést nyerhessen rendszer fájlokhoz és kicserélje azokat kártékony kódot tartalmazó fájlokra, vagy kihasználva egy webalkalmazás gyengeségeit, amely következtében a bizonytalanul telepített háttéradatbázis adatai módosíthatóak vagy törölhetőek.

## 8.4.5 Incidenskezelési folyamat

Az incidenskezelési folyamat számos lépésből állhat, kezdve az inicializálási előkészületektől az incidensek utóelemzéséig. Minden egyes fázis fontos szerepet tölt be, és alaposan definiálandó, követendő a szervezeten belül annak érdekében, hogy a megfelelő biztonsági szint elérhető legyen.




*35. ábra Az incidensek kezelésének folyamata*

A legfontosabb fázisok:
- előkészítés,
- detektálás és elemzés,
- elhatárolás, felszámolás és helyreállítás,
- utó-incidens tevékenységek.

## 8.4.6 Problémamenedzsment

Az incidens menedzsment célja a szolgáltatások és üzleti műveletek visszaállítása amilyen gyorsan csak lehet, míg a problémamenedzsment a szolgáltatások és üzleti műveletek fejlesztésére fókuszál. Amikor egy incidens oka ismeretlen, problémáról beszélünk.

A problémamenedzsment célja, hogy meghatározza és megszüntesse a probléma kiváltó okát, ezzel javítva az üzleti műveleteket biztosító folyamatokat. A problémamenedzsment egy fontos feladata, hogy miután fixálta a beazonosí-

tott kiváltó okot, biztosítja, hogy a probléma többet nem fordul elő. Az ismétlődő incidensek elkerülése az egyike a két kritikus sikertényezőnek a problémamenedzsmentben. A második a kedvezőtlen hatások minimalizálása az üzleti folyamatokra.

A problémamenedzsment bejelentéssel kezdődik, és riport készítésével ér véget:


*36. ábra Problémamenedzsment folyamata*

## 8.4.7 Hibakezelő adatbázisok

A kész üzleti alkalmazások és rendszerek tovább sérülhetnek, ahogy újabb fenyegetéseket fedezünk fel. Ekkor szükség van a támadások által okozott sebezhetőségek javítására. Ebben a helyzetben a szoftvert nem távolítjuk el teljes egészében, hanem hozzáadunk a kódhoz, amelyek a sebezhetőségek javítását és implementálását célozzák meg. Ezek a hozzáadott részek, melyeket patch-nek is nevezünk, frissítésként vagy fixálásként szolgálnak a létező szoftverben, így a továbbiakban a rendszer nem lesz érzékeny ismert bugokra.

A patchek megvalósításának több módja is van:

- hotfix vagy Quick Fix Engineering (QFE):
- szervizcsomag (Service pack):

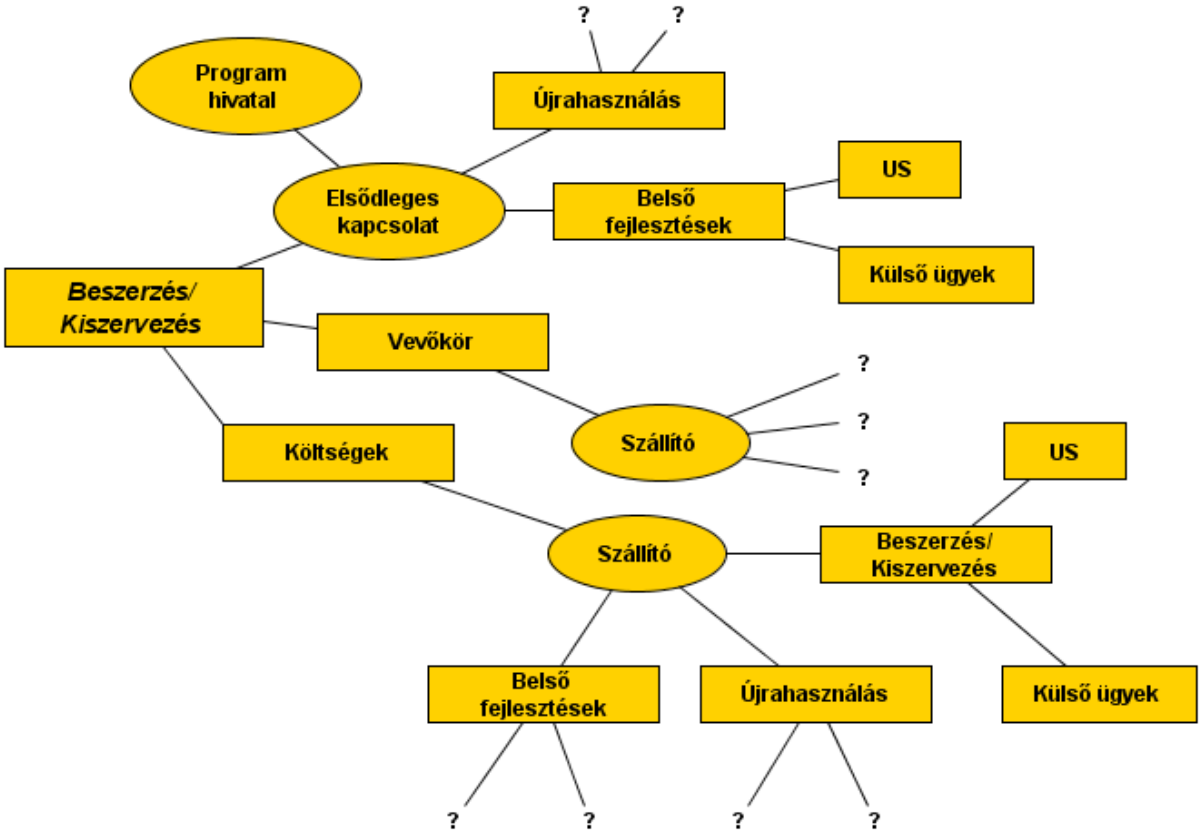
A **hotfix** egy funkcionális vagy biztonsági patch, amelyet a szoftver fejlesztőjének vagy gyártójának kell előállítani. Gyakran előfordul, hogy nem tartalmaz új funkcionalitást vagy tulajdonságot, valamint változtatásokat a hardverben és a szoftverben. A hotfixek általában az operációs rendszerhez kapcsolódnak vagy valamilyen platformkomponenshez. A hotfixek előnye, hogy a javítás egyesével vagy szelektíven is elvégezhető.

A **szervizcsomag** általában több hotfixet tartalmaz egy csomagban, melyek olyan frissítéseket tartalmaznak, amelyeket ismert hibák javítására készítenek, valamint tartalmazhatnak további fejlesztéseket és funkciókat is. A Periodikus szoftverfrissítéseket gyakran publikálják szervizcsomagokban, valamint az új termékverzióknak tartalmazniuk kell a korábbi szervizcsomagok tartalmát, illetve együtt kell működniük minden eddig kiadott szervizcsomaggal, hogy megbizonyosodjunk róla, hogy nincsenek regressziós eredmények, különösen amelyek a biztonsághoz kapcsolódnak. A szervizcsomagok előnye, hogy egyszerre több hotfixet tud alkalmazni, amely idő- és költséghatékonyabb, mintha egyesével telepítenénk a hotfixeket.

***Fogalmak a 8.4. fejezetben:*** incidens, esemény, figyelmeztetés, Incidens Menedzsment, patch, hotfix, szervizcsomag, problémamenedzsment.

## 8.5 Szerződések szerepe

A szoftver forráskódja, valamint programkódja és a hozzá tartozó dokumentáció a programozók szellemi alkotása. Mindezen alkotások szerzői jogával a szoftver alkotója rendelkezik. A szoftver létrejöttének pillanatától szerzői jogvédelem alatt áll. A szerzői jogról lemondani nem lehet, nem eladható, másra át nem ruházható. A szerzői jogvédelmi törvény (1999. évi LXXVI. törvény) alapján a szoftvert a szerzői jogvédelmi ideje alatt csak fizetés ellenében szabad felhasználni.




*37. ábra A szoftver ellátási folyamat lehetséges útvonalai*

A szoftveralkotások felhasználására licenszek vásárlásával szerezhetünk jogot. A licenszek megvásárlásával a szoftver kiadója feljogosítja a vevőt a termék használatára, a vevő pedig ezzel (illetve számlával, szerződéssel) igazolja annak származását. Tehát a vevő a licensszel nem a szoftver (másolásra és továbbadásra feljogosító) tulajdonjogát, hanem csak a használati jogát kapja meg.

### 8.5.1 Demóprogramok, shareware, freeware, adware

A szerződéseknek, licenceknek több fajtája lehet, amelyek a használat többféle típusát teszik lehetővé. Ilyenek a

- demóprogramok
- shareware
- freeware
- adware

A **demóprogramok** gyakran olyan alkalmazásokat jelentenek, amelyek egy nagyobb programba engednek kisebb betekintést. Gyakran csak adott részeit tartalmazzák a programnak, annak minden funkciójával, így a felhasználó eldöntheti, hogy a rendszert a teljes funkcionalitásával kívánja használni, és hajlandó-e fizetni érte.

A **shareware** licencű programok általában ingyenesen kipróbálható, „próbáld ki, mielőtt megveszed" címkéjű szoftverek, amelyek a teljes funkcionalitás kipróbálását biztosítják egy bizonyos ideig – általában 15, 30 napig. Amint a felhasználó kipróbálta a programot eldöntheti, hogy az megfelel-e az igényeinek, és érdemes-e fizetni érte. Ha továbbra is használni szeretné a felhasználó a programot a próbaidő lejárta után, fizetnie kell érte, hogy azt korlátlan ideig használhassa. Amennyiben a felhasználó tetszését nem nyerte el a program, egyszerűen csak uninstallálnia kell azt.

A **freeware** programok teljesen ingyenesen használhatóak és terjeszthetőek. Ezek a programot több célból is lehetnek ingyenesek: pusztán emberségből, hírnév szerzéséért, mint reklám, vagy lehet egy program kereskedelmi változatának lecsupaszított, leegyszerűsített verziója.

Az **adware** programok (reklám által támogatott szoftver) olyan termékek, melyek ingyenesen használhatóak, cserébe reklámokat és hirdetéseket jelenít meg a számítógépen.

## 8.5.2 Szabad szoftverek, nyílt forráskódú szoftverek, egyéb FLOSS típusok

A **szabad szoftverek** bárki által használható és módosítható szoftverek, amelyek a következő szabadságjogokkal kell, hogy rendelkezzenek:

- szabad felhasználás tetszőleges célra
- szabad tanulmányozhatóság és igény szerinti módosíthatóság, amelynek előfeltétele a forráskódhoz való hozzáférhetőség,
- másolatok szabad terjeszthetősége,
- szabad továbbfejleszthetőség, valamint annak szabad közzététele.

Fontos megemlíteni, hogy a szabad szoftverek nem feltétlenül ingyenesek: a szabad szoftverek értékesíthetők is, a lényeg, hogy a fent említett négy szabadságjogot biztosítani kell a vásárlók számára.

A **nyílt forráskódú licensz**eket az OSI véleményezi és tartja nyilván. A nyílt forráskódú licenszek között megtalálni a FSF (Free Software Foundation) GNU licenszeit, de a Microsoft Ms-PL és Ms-RL licenszeket is. A nyílt szoftverek licenszei nem minden esetben kompatibilisek: előfordulhat, hogy két szabad licensszel rendelkező szoftvercsomag forráskódját nem lehet kombinálni és terjeszteni a licenszek összeférhetetlensége miatt.

Az FSF álláspontja szerint a nyílt forráskódú szoftver nem tökéletes szinonimája a szabad szoftvernek, mivel a megnevezés nem hangsúlyozza ki a felhasználó szabadságát. A **FLOSS** szinonima feloldja a különbséget azzal, hogy a Free/Libre/Open Source Software kifejezés rövidítésével a szabadságot és a nyílt forráskódot is kellőképpen hangsúlyozza a megnevezésben, rámutatva arra, hogy lényegi különbség nincs a szabad és a nyílt forráskódú szoftverek között.

## 8.5.3 Licencszerződés tipikus tartalma

A szoftver licenc egy hivatalos, jogi eszköz, amely a szoftver használatát és terjesztését szabályozza. Egy tipikus szoftver licenc garantálja a végfelhasználói engedélyt arra, hogy a szoftver egy vagy több példányát használja olyan módon, hogyha a licencben foglaltakat a felhasználó megszegi, annak komoly jogi következményei lehetnek.

A szoftver licenszek tipikusan a következőket tartalmazzák:

- rendelkezések, amelyek a kötelezettségeket és felelősséget írják le a felek között,
- felelősség korlátozása,
- garanciális nyilatkozat,
- kártérítési folyamat.

A licensz szerződés megsértője törvénysértést követ el. Az illegális szoftverhasználat, a szoftver hamisítása, ugyanazon program több gépre telepítése jogosulatlanul, internetről letöltött illegális szoftverek használata szoftverkalózkodásnak számít.

## 8.5.4 Beszállítói életciklus

Az ellátási láncot fel lehet bontani különböző fázisokra az életcikluson belül. Ezek a fázisok név szerint a következők:

- tervezés,
- szerződéskötés,
- fejlesztés és tesztelés,
- elfogadás,
- telepítés,
- üzemeltetés és monitorozás,
- átmeneti időszak,
- kivezetés.


*38. ábra Ellátási lánc fázisai*

A **tervezési** fázisban elkészül a kezdeti kockázatelemzés annak érdekében, hogy meghatározhassuk a funkcionális és minőségbiztosítási igényeket, követve az ellátási stratégiát vagy tervet.

A **szerződéskötés** fázisa magában foglalja a hirdetés kiadását a beszállítók számára, a szállítók értékelését és a válaszokat, a szerződési tárgyalásokat, a beszállítók kiválasztását, valamint a szerződések odaítélését.

A **fejlesztés és tesztelés** fázis magában foglalja a megbízható, rugalmas és újrahasznosítható kód implementációját, és a biztonsági ellenőrzések hitelesítését.

Az **elfogadási** fázisban megtörténik az elfogadási kritérium definiálása, az egyes tevékenységek validálása és verifikálása, beleértve a független tesztelést, a felvásárlói elfogadást és a szerződés véglegesítését.

A **szállítás** magában foglalja a kóddal kapcsolatos szerződések létrehozását, a kommunikációt és teljesítmény ellenőrzését, beleértve a biztonságos adatátvitelt.

Az üzemeltetés és monitorozás magában foglalja a szerződési munkarend érvényesítését, a telepítés utáni támogatás, a változás- vagy konfiguráció-menedzsment ellenőrzési eljárásainak létrehozását, stb.

A kivezetés magában foglalja azokat a tevékenységeket, amelyek segítenek az információt fenyegető kockázatok enyhítésére vagy elkerülésére. Ebben a fázisban a szoftver „leszerelésre” kerül, leállítva a szoftvert és a hozzá tartozó folyamatokat.

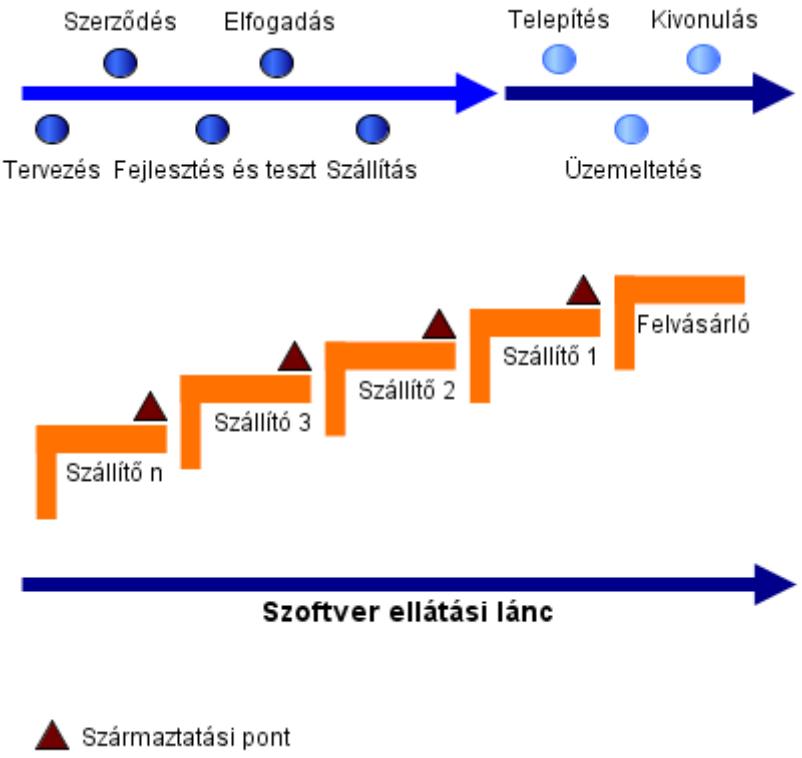
## 8.5.5 Beszerzés fajtái

Egy szoftver beszerzésére több módszer is létezik: közvetlen beszerzés, Original Equipment Manufacturer (OEM) licenszelés, partnerkapcsolat létrehozása a szoftver gyártójával, outsourcing és menedzselt szolgáltatások.

## Outsourcing

A szoftver kihelyezés magában foglalja a szoftverfejlesztő alvállalkozóval és a kapcsolódó szolgáltatókkal történő közreműködést. A szolgáltatásokat és fejlesztési tevékenységeket részekre bontja, amelyeket alvállalkozók teljesítenek a lehető leghatékonyabb és költséghatékonyabb módon. A külső beszállító lehet egy szoftverfejlesztő cég, amely lehet hazai vagy külföldi.

Mivel a szoftver az egyik szolgáltatótól a másik kezébe kerül, a védelem felelőssége is továbbhárul az egyik szolgáltatótól a másikig. A felelősség eltolódásának és a felügyelet elvesztésének jelenségére szoftvereredetként hivatkozunk.




*39. ábra Szoftver ellátási lánc a felelősség átadásával*

A **menedzselt szolgáltatások** lehetővé teszik az erőforrás-igénylést intenzív üzleti műveleteknél, majd azokat tapasztalt szakemberek kezébe adják menedzselésre. Ezek a menedzselt szolgáltatások magukban foglalhatják a nem biztonsági jellegű szolgáltatásokat, mint pl. a szoftverfejlesztés és előfizetéses szolgáltatások, hogy aztán biztonságos szolgáltatásokat specifikáljanak, mint pl. kockázatmenedzsment, biztonsági kockázatkezelés, behatolás tesztelés, incidens menedzsment, vírusirtás és tartalomszűrő rendszerek.

Az As-a-Service megoldások, a Platform-as-a-Service (PaaS), az Infrastructure-as-a-Service (Iaas) és a Software-as-a-Service (SaaS) egy-egy példa a menedzselt szolgáltatások kidolgozott megoldásaira, amelyek a mai napjainkban a leguralkodóbb módszerek a témában.

## A beszerzői láncok kockázatai

Egy támadás, amely a beszerzői láncban található biztonsági réseket célozza meg és használja ki, ellátási lánctámadásnak nevezzük. A szoftvertermékek, azok által nyújtott szolgáltatások, egyedi termékek vagy beágyazott szoftverek a hardverben mind ki vannak téve az ellátási lánc-támadásoknak. A legpotenciálisabb és uralkodó fenyegetés a szoftver ellátási láncban, amely gyakran észrevétlen marad, az a szoftver kódjába történő beavatkozás rosszindulatú szoftverek bevezetésével, mialatt a szoftver fejlesztése történik.

A beszerzési lánc hátránya, hogy további számos támadás veszélyeztetheti a szoftvert, annak szolgáltatásait, annak folyamatait és az embereket is. Ilyen hibák lehetnek a következők:

**Termék- és adatveszélyeztetések:**

- beavatkozás kódrészekbe, hogy ezzel kerüljék meg a biztonsági ellenőrzéseket,
- jogosulatlan közlések, módosítások, adatok megsemmisítése vagy megszerzése,
- kód szabotálása, szándékos sérülékenység és rosszindulatú logika beültetésével, stb.

**Folyamat sérülékenységek:**

- legitim folyamatok kikerülése és burkolt eltérítések törvényes csatornán történő véghezvitele,
- export kontrollkövetelmények megsértése,
- nem biztonságos kód átadása, amelyeket nem kezel a beszerzési lánc, stb.

**Emberi fenyegetések:**

- fel nem ismert rosszindulatú fenyegetések jelenléte, (pl. hacker, bűnöző, ellenség) a cégen belül. Ezt a jelenséget belső fenyegetésnek szoktuk nevezni.
- belső munkatársak, akik csalást vagy hamis tanúzást követnek el (pl. felbujtók, hamis tanúvallomásra bírók), stb.

## 8.5.6 Kockázatmenedzsment

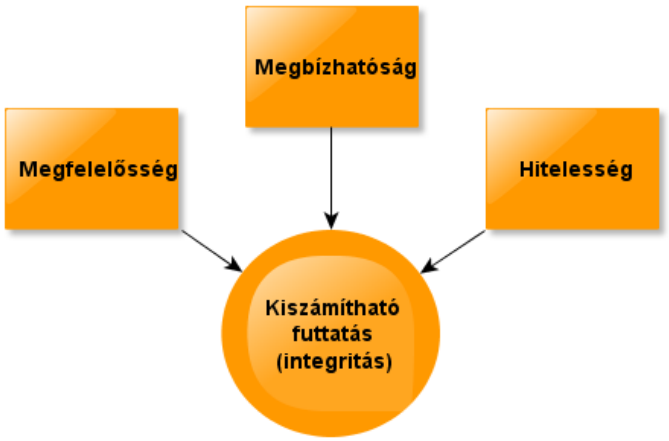
Egy gyengeség vagy üzemzavar bármely folyamatban az ellátási láncon belül kritikus lehet az egész folyamatra nézve. Ezeket a kockázatokat az ellátási lánc bármeny komponensén észlelhetjük, valamint ezek öröklődhetnek további alrészekre.

A hagyományos metódusok a biztonsági rések csökkentésére, biztonsági fejlesztési praktikákat használva, ha szükséges, kívül esik a megbízhatóság biztosításán, amely azt garantálja, hogy a szoftver hitelesen és megbízhatóan működik.

A szoftver helyes működése ellen fellépő kockázatok, melyek fenyegetésekből fakadnak, több úton illeszkedhetnek be az ellátási láncba, akár a fejlesztési vagy tesztelési, vagy működtetési és karbantartási fázisban. A kockázatok érvényesülésére következők a lehetőségek:

- szállítók elégtelen validálása vagy beszerzési módszere,
- a szerződési nyelvet nem vették figyelembe a biztonsági követelmények kidolgozásánál,
- véletlen tervezési defektusok és kódolási hibák, melyek kihasználhatják a sebezhetőséget,
- rosszindulatú logika beillesztése arra nem jogosult felhasználóktól a szoftver kifejlesztése után,
- helytelen kódpublikálási folyamatok, melyek nem tisztázzák a kód származásának hitelességét,
- hiba a fenyegetések és patch-ok kezelésének menedzsment-folyamataiban, amelyek kockázatokkal járhatnak, stb.

Az Ellátási Lánc Kockázat Menedzsment feladata ezen kockázatok elhárítása és kezelése, amelyet a következő ábra mutat be:




*40. ábra Szoftverellátási lánc, kockázatmenedzsment céljai*

## 8.5.7 Beszállítók szerződései, szerződés átruházása

A beszállítók és a felvásárlók közötti kapcsolat, valamint a különböző szállítók közötti kapcsolat hatással van a kockázatok és a vezérlés szintjére. A szoftverellátási láncban két elsődleges kapcsolattípusról beszélhetünk: work-for-hire (alvállalkozók szerződtetése vagy személyzet bővítése) és licenszelő kapcsolatokról.

A felvásárolók a következő lehetőségekkel rendelkeznek:

- A szoftver fejlesztésével kapcsolatban szerződést kötnek más beszállítókkal. Ez a fajta kapcsolat „alvállalkozó bérlése" kapcsolatnak nevezzük, ahol a felvásárló birtokolja a szoftver szállításának jogát.
- Más szállítók személyzetével történő együttműködés. Ezt a fajta kapcsolatot „személyzethosszabbító munkabérlésnek" nevezzük.
- Egy másik beszállítótól megvásárolni egy szoftver engedélyét, vagy egy nyílt forráskódú szoftver megszerzésére törekedni. Ezen típusú kapcsolat neve „engedélyek közelebb hozása".

Fontos megjegyezni, hogy minden egyes beszállító a beszállítási láncban állhat hasonló kapcsolatban más beszállítókkal.

### Szolgáltatás szintű megállapodások (SLAs)

Az SLAk formális megállapodások egy beszállító és a beszállító termékeinek és/vagy szolgáltatásainak vásárlója között. Ezek a dokumentumok tartalmazhatnak jogosultságokat és követeléseket azokban az esetekben, mikor inkább szerződésként kezelik az ilyen típusú megállapodásokat. Beszállítók kiválasztásánál egy szállító SLA felülvizsgálata elengedhetetlen annak érdekében, hogy meggyőződjünk arról, hogy a szállító képes együttműködni a biztonsági funkcióban az általuk fejlesztett szoftvertermékkel kapcsolatban, valamint támogassák és tartsák karban termékük az adatrögzítések után is.

Az SLA-k két típusát különböztetjük meg a követelmények tekintetében: lehetnek

- követelmény-függők
- és követelmény-alapúak.

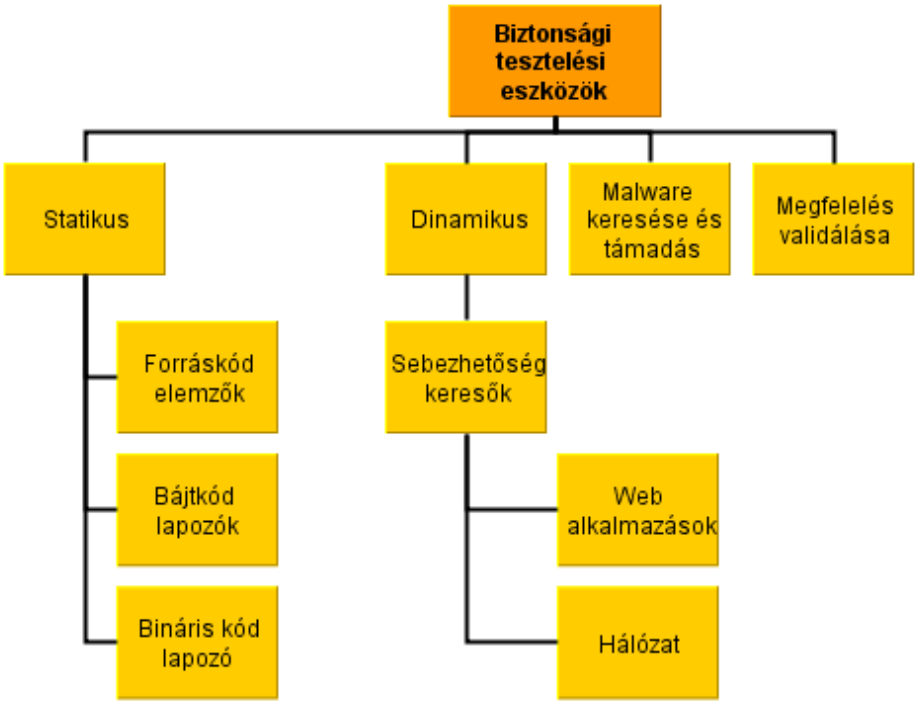
A **követelmény-függő SLA**-k olyan megállapodások, melyeknek tartalmaznia kell az üzleti követelményeket; pl. egy bérszámfejtési rendszer nem szenvedhet üzemzavaroktól, míg ez egy nem-kritikus humán erőforrás tréning rendszer esetében nem jár katasztrofális következményekkel.

Egy **követelmény-alapú SLA** olyan típusú szerződéseket takar, melyek a pontosan definiált követelmények és a tényleges szolgálati szintek meghatározása nélkül pontatlan szerződést eredményeznek.

## 8.5.7 Beszállítók kiértékelése. Beszállítók által készített tipikus termékek kiértékelése

A szoftverellátási lánc kockázatmenedzsmentje a tervezéstől kezdve egészen a kiszállításig tart. Azok a tevékenységek, amelyek a fejlesztési és tesztelési fázisban kerülnek sorra, nagyobb figyelmet élveznek, mivel a kártékony résztvevők, kiknek hozzáférése van a kódhoz, kártékony logikát és kódot szúrhatnak be a rendszerbe, melyeket nehéz felfedezni és komoly károkat okozhatnak a szoftver kifejlesztése után, ha az nincs kellően tesztelve biztonsági vonatkozásban. A biztonsági intézkedések magukban foglalják a követelmények megfelelőségének validációját, a kódfelülvizsgálatot, a hozzáférési ellenőrzéseket kód repository-khoz, a felépített eszközök és környezet integritását és a biztonsági tesztelést.

A biztonsági tesztelési eszközök a következők lehetnek:




*41. ábra Biztonsági teszteszközök*

***Fogalmak a 8.5. fejezetben:*** demóprogramok, shareware, freeware, adware, szabad szoftver, nyílt forráskódú szoftver, licenc, outsourcing, Original Equipment Manufacturer, Platform-as-a-Service, Software-as-a-Service, kockázatmenedzsment, SLA.

# FELHASZNÁLT IRODALOM

IEEE829 szabvány
IEEE830 szabvány
ISO9126 szabvány
ISO25000 szabvány
International Software Tesing Qualifications Board: Certified Tester Foundation Level Syllabus
International Requirements Engineering Board: Certified Professional for Requirement Engineering Foundation Level
Magyar Szoftvertesztelői Tanács Egyesület: Szoftvertesztelés egységesített kifejezéseinek gyűjteménye.
Mano Paul (editor): OFFICIAL (ISC)2 GUIDE TO THE CSSLP CBK (second edition). e-book, CRC Press 2014

Nemzeti
Közszolgálati Egyetem
Vezető- és Továbbképzési Intézet

LEITOLD FERENC

# Sebezhetőségvizsgálatok a gyakorlatban

Budapest, 2014

A tananyag az ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel című projekt keretében készült el.






A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

**Szerző:**
© Leitold Ferenc 2014

**Kiadja:**
© NKE, 2014

**Felelős kiadó:**
Patyi András
rektor

# TARTALOM

# BEVEZETÉS

A számítógépes hálózatokon keresztül történő, a számítógépek és a felhasználók kommunikációját kihasználó támadások, fenyegetések egyre nagyobb veszélyt jelentenek. Ide tartoznak az interneten terjedő kártevők, a célzott támadások botnet hálózatok igénybevételével vagy anélkül, és ide sorolhatjuk a social engineering alapú, a személyes kommunikációra épülő támadásokat is.

A kommunikációs csatornák biztonsága két kérdéskört érint. A középkorban a futárral történő üzenetküldés legnagyobb kockázatát az út során történő információszerzés jelentette. Ezért az információt titkosították, rejtjelezték. Egy támadó két támadási módszer közül választhatott. Mint passzív támadó lehallgatja az üzenetet, megfejti és saját céljaira felhasználja, de NEM akadályozza az eredeti üzenet célba érkezését. Másrészt, mint aktív támadó megtehette azt is, hogy a lehallgatott információt módosítva küldi tovább, esetleg válaszol az eredeti üzenet küldőjének.

A középkorban még nem volt jelentős az a probléma, amit az jelent, hogy a támadó ellenőrzése alá vonja az egyik végpontot (küldő vagy fogadó). Manapság azonban a csatorna lehallgatása mellett ez egy sokkal jelentősebb veszélyforrást jelent. A kommunikációs eszközök, az internet gyors fejlődésével egy támadó valós időben felügyelheti a megtámadott eszközeit, illetve saját céljainak megfelelően befolyásolhatja működésüket.

Ebben a tananyagban a számítógép-felhasználókat leginkább fenyegető, egyrészt a technikai eszközöket támadó, másrészt a felhasználókat érintő, social engineering alapú veszélyforrásokat tekintjük át. A legelterjedtebb veszélyforrásokra vonatkozóan azon vizsgálati módszerek is ismertetésre kerülnek, amelyek célja ezen sebezhetőségek felderítése.

# 1. INFORMATIKAI RENDSZEREK TÁMADÁSI LEHETŐSÉGEI

Az internet elterjedésével az informatikai rendszerek biztonságának egy sarkalatos kérdésévé vált azok biztonsága az internetről érkező támadásokkal szemben**. A hálózatbiztonság témakörébe tartozik minden olyan biztonsági kérdés, amin nem javíthatunk csupán titkosítással.** Manapság a legtöbb informatikai rendszert ért támadás az internetről származik. Az 1990-es évektől az internet rohamos fejlődése a XXI. század első évtizedének a végére egy termékeny melegágyat teremtett az informatikai támadásoknak. Már 2008-ban a támadások 92%-a az internetről érkezett és csupán 8% vett igénybe valamilyen adathordozót a bejutáshoz. Az internetről érkező veszélyeket alapvetően két csoportba sorolhatjuk. Egyrészt az interneten elérhető és onnan érkező kártevők köre, másrészt a célzott támadásokkal kapcsolatos biztonsági kérdések. A célzott támadások esetén a támadó kifejezetten a megtámadott informatikai infrastruktúrába szeretne behatolni, hogy ott a felhasználók tudta nélkül tevékenykedjen. Ebben a fejezetben a teljesség igénye nélkül a támadási formák fejlődését tekintjük át, majd kiemelten foglalkozunk az internetről érkező veszélyekkel is. Az egy informatikai rendszert potenciálisan veszélyeztető célzott támadási lehetőségekkel, a legújabb APT támadási módszerekkel, valamint a hálózati támadások eljárásaival egy-egy alfejezetben külön foglalkozunk.

## 1.1 A támadási formák fejlődése

A számítógépes kártevők alapötlete az 1940-es évekre vezethető vissza. Neumann János ugyanis nemcsak a mai számítógépek (tárolt programú automaták) működését tervezte meg, hanem az "Önreprodukáló automaták elmélete" című tanulmányában a terjedésre képes számítógépes kártevők működését is leírta ([5]).

A szobányi területet elfoglaló nagygépes rendszereken jelentek meg az első féregprogramok. (Az egyik első ilyen kártevő a *Morris féreg* volt az 1970-es években.) Az operációs rendszer réseit használták ki, viszont főként csak az adott nagygépes rendszerben voltak képesek terjedni, hiszen akkoriban nem volt kapcsolat ezen számítógépek között. Céljuk általában az információszerzés (például jelszótáblák) volt. Nem irtották őket, ehelyett inkább kijavították az operációs rendszer hibáit, így nem tudtak elterjedni.

Az első IBM PC kompatibilis számítógép megjelenését követő néhány éven belül, 1986-ban jelent meg az első PC-s vírus, a Brain, amely floppy lemezek, illetve a merevlemez boot szektorát fertőzte meg. Néhány évvel később Magyarországon is megjelentek az első vírusok: 1988 őszén bukkantak fel a Cascade, illetve a Vienna programvírusok első változatai.

A 90-es évek elején a legelterjedtebb operációs rendszer a DOS volt, a Windows programrendszert is DOS alól kellett indítani. A Microsoft 1995-ben jelentette meg a DOS nélküli Windows operációs rendszert, illetve ekkor jelent meg az Office programrendszer is. A kártevők készítői hamar felfedezték az új rendszerekben rejlő lehetőségeket, és új irányokat kerestek. 1994-ben jelent meg az első Word alatti makróvírus kísérleti példánya (DMV), illetve a következő évben a Concept makróvírus fertőzte végig a világ számítógépeinek a nagy részét.

A makróvírusok megjelenéséhez hasonló áttörést jelentett az Internet megjelenése is. A 2000-es évektől már a legtöbb fertőzést az interneten terjedő kártevők, elsősorban férgek (például Lovegate, Melissa) jelentették. A férgeken túlmenően egyre inkább tért hódítottak a Phishingek, spamek, HOAX-ok is.

Az Internet nem csupán a terjedés eszköze lett, hanem egyre inkább a támadások célpontját is jelentette. A kártevők készítői hamar felismerték, hogy hatékonyan vihetik végbe támadási törekvéseiket, ha a megtámadott számítógépekből hálózatot (botnet) alakítanak ki, és e számítógépeket együttesen veszik igénybe a támadás céljának megfelelően. Az Internet ugyanakkor egyre bővülő reklámfelületet jelentett és jelent, amit spam üzenet küldésére, reklámablakok megjelenítésére használnak ki. A nagy volumenű támadások mellett ugyanakkor egyre kifinomultabb módszereket használnak specializált kémtevékenységre is.

Egy futtatható kártékony kódot tartalmazó kártevő célja minden esetben az, hogy a kártékony kód a megtámadandó számítógépen végrehajtásra kerüljön. A számítógépek felépítése szerint a futásra kerülő programkódnak a memóriában kell lennie és a processzornak végre kell hajtania azt. Egy kártevőt a számítógép memóriájából a legegyszerűbben a kikapcsoló gomb használatával távolíthatjuk el. Ekkor ugyanis a számítógép memóriájának tartalma, és ezzel együtt a kártevő kódja is törlődik. A kártékony programoknak – annak érdekében, hogy működőképességüket megőrizzék – az a céljuk, hogy képesek legyenek arra, hogy a későbbiekben (például a számítógép következő elindulásakor) aktivizálódjanak. Ennek érdekében a futtatható kódot tartalmazó kártevőknek két lényeges feladatot kell el-

végezniük. Egyrészt a saját programkódjukat olyan helyre kell tenniük, aminek a tartalma a számítógép kikapcsolását követően is megmarad, másrészt gondoskodniuk kell arról, hogy ez a programterület a számítógép későbbi működése során, automatikusan vagy valamilyen esemény hatására végrehajtásra kerüljön.

A számítógépes kártevők egy jelentős csoportját képviselik azok a kártékony programok, melyek képesek arra, hogy önmagukat másolva szaporodjanak. Amennyiben egy támadó nem célzott támadást szeretne végrehajtani, az önmagát másoló mechanizmust tudja kihasználni annak érdekében, hogy a kártékony kód sok számítógépre eljusson. Ilyenkor a kártékony program kódja általában két fő részből áll: egyrészt a programkódot megvalósító részből, másrészt pedig a kártékony tulajdonságot megvalósító programrészből. Ez utóbbi nem kötelező része a kártékony kódnak, csupán opcionális. Az önmagukat másoló, szaporodásra képes kártevők egy jelentős része ugyanis a szaporodáson kívül semmi egyebet nem tesz. Ez viszont nem jelenti azt, hogy az ilyen programkódok ne lennének kártékonyak. A szaporodási eljárás ugyanis önmagában kártékony, hiszen igénybe veszi a számítógép erőforrásait.

Az önmagukat másoló, szaporodást végző kártevők két csoportját különböztethetjük meg aszerint, hogy szaporodásukhoz szükséges-e hordozóprogram, vagy sem. A hordozó programot igénylő kártevőket *vírusoknak*, a hordozóprogram nélkülieket pedig *férgeknek* nevezzük.

# 1.2 Interneten terjedő kártevők

Az Interneten terjedő kártevők célja, hogy egy másik számítógépre vagy egy másik felhasználóhoz jussanak el. Ezek a kártevők tehát üzenet küldésén alapulnak, az üzenet célpontja lehet egy felhasználó vagy lehet egy alkalmazás.

Amennyiben a kártevő terjedéséhez használt üzenet célpontja a felhasználó, akkor a kártevő általában egy üzenetküldési szolgáltatást használ, mint például e-mail, Skype, MSN, ICQ, IRC vagy akár a közösségi oldalak (például Facebook), illetve a csoportmunkára használt tárhely-szolgáltatások (például Dropbox) megosztásait. A felhasználóknak címzett üzenetek két fő csoportját különböztethetjük meg:

Ha **az üzenet tartalmaz** olyan **programkódot**, amely a számítógép számára bármilyen módon értelmezhető, akkor a kártevő célja az, hogy ez a programkód a másik számítógépen végrehajtódjon. A kártevő ilyenkor valamilyen módon megpróbálja rávenni a felhasználót arra, hogy az átküldött kódot végrehajtsa (rákattintson a mellékletre). Ehhez többféle, social engineering-en alapuló módszert használhat:

- A csatolt állomány nevében a dupla kiterjesztés használatával elérheti, hogy a Windows alapértelmezés szerint elrejtse a fájl valódi típusát.
- A csatolt fájl nevében a valódi kiterjesztés elé sok szóköz karakter elhelyezésével a Windows nem képes megjeleníteni az állomány teljes nevét és a valódi kiterjesztés lemarad.
- Webcímre emlékeztető nevet használ: például a www.myparty.com. Ha ezt egy csatolmány nevében látjuk, akkor az nem egy webcím, hanem egy .COM kiterjesztésű állomány.
- Az üzenet feladójának meghamisításával elérheti, hogy úgy tűnjön, mintha az üzenet a címzett valamely ismerősétől érkezett volna.
- Az üzenet szövegével vagy a csatolmány nevével felkelti a felhasználó figyelmét, érdeklődését.

A gyanútlan felhasználónak címzett üzenetek ugyanakkor az említett social engineering alapú, a felhasználó mint emberi tényező átverésére irányuló technikák mellett megcélozhatják az üzenetet fogadó vagy kezelő alkalmazást is. Ezen alkalmazások valamely biztonsági problémáját kihasználva elérheti, hogy az alkalmazás automatikusan végrehajtsa az elküldött mellékletet.

Ha **az üzenet nem tartalmaz** semmilyen **programkódot**, amely a számítógép számára bármilyen módon értelmezhető lenne, akkor a kártevő célja az, hogy a felhasználó valamilyen tevékenységet hajtson végre. Ilyen tevékenységek általában az alábbiak:

Ha az üzenet egy webcímet tartalmaz, akkor a kártevő célja, hogy a felhasználó a webhelyet nyissa meg a böngészőjében. Ez továbbviheti a felhasználót például egy olyan weboldalra, ahonnan kártékony kód települ a számítógépére, vagy egy phishing oldalra, ahol megpróbálják a személyes adatait megismerni.

Ha az üzenet nem tartalmaz webcímet, akkor a kártevő célja az lehet, hogy a felhasználó olvassa el az üzenetet (kéretlen reklámüzenet, SPAM) vagy, hogy a felhasználót használja a terjesztés "motorjaként" (lánclevél, HOAX).

Amennyiben a cél egy alkalmazás, akkor a kártevő az alkalmazás egy biztonsági rését használja ki. Ilyenkor a biztonsági rés révén a kártékony kód automatikusan elindul. A biztonsági rés kapcsolódhat az *operációs rendszerhez*, az operációs rendszeren futó *alkalmazáshoz*, esetleg valamely *protokollhoz* is. A biztonsági rések egy szűkebb körét valamely fájl-formátumhoz tartozó sérülékenységnek is szokták nevezni, (a teljesség igénye nélkül) például az SWF, JPG vagy MP3 fájlok esetén. Ilyen esetekben azonban a probléma általában NEM magához a fájl formátumához kapcsolódik, hanem az adott fájl-formátumot kezelő alkalmazáshoz.

Az interneten terjedő kártevők esetén elmondhatjuk, hogy a hatásos terjedésük két fő tényezőre vezethető vissza. Terjedésükhöz egyrészt az *emberi tényező*, másrészt a *biztonsági rések* is hozzájárulnak. A biztonsági rések viszont szintén az emberi tényezőre vezethetők vissza. Egyrészt a biztonsági rések a programok, alkalmazások fejlesztése során, a nem megfelelő tervezésből, programozásból, kódolásból adódnak. Másrészt a biztonsági rések nagy részét a megfelelő, és időben végrehajtott frissítésekkel el lehet kerülni. Így leszűrhetjük azt a végkövetkeztetést, hogy az interneten terjedő kártevők terjedése elsősorban az emberi tényezőre vezethető vissza.

## 1.3 Célzott támadások

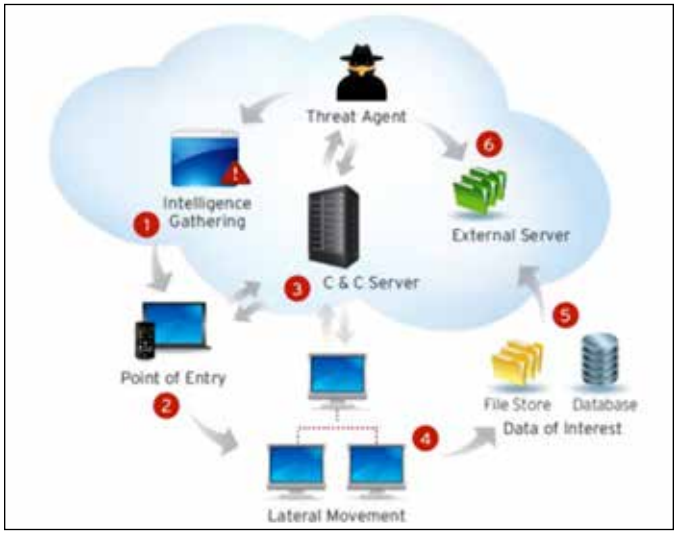
Célzott támadásoknak nevezzük az olyan fenyegetéseket, melyeket a támadók kifejezetten egy adott célpont (személy vagy szervezet) ellen használnak. Egy számítógépes vírushoz képest a fenyegetés „megalkotója" ebben az esetben nem arra törekszik, hogy a kártékony kód minél jobban elterjedjen, hanem arra, hogy a kiszemelt célpont eszközére, eszközeire bejusson.

Célzott támadások már a 2000-es évek elején is léteztek, sőt néhány esetben Magyarországon is megjelentek. Az egyik ilyen hazai esetről szóló esettanulmány ([1]) szerint 2001-ben egy magyarországi – mintegy 200 számítógéppel rendelkező – intézmény vált célzott támadás áldozatává. A tanulságos történet szerint a vezetés a rendszergazda elbocsátását követően néhány hónappal vette észre, hogy kezdik sorozatosan elveszteni a tendereket. Megbíztak egy informatikai biztonsággal foglalkozó vállalkozást a szervezet átvilágítására, amelynek során a számítógépeket is szúrópróbaszerűen megvizsgálták. Miután a második olyan számítógépet is felfedezték, amely egy sehova nem köthető kis programot tartalmazott, a vizsgálatot kiterjesztették a szervezet valamennyi számítógépére. Összesen 11, a szervezet működéséhez szükséges legfontosabb számítógépen került elő ez a kis program, beleértve a vezetők hordozható számítógépeit is. A kis program elemzése során kiderült, hogy az alkalmas arra, hogy információkat küldjön a szervezeten kívülre, másik számítógépre telepedjen. Mintegy 16 különböző módszerrel rendelkezett a külső kommunikáció érdekében, a kapcsolatot pedig számtalan úgynevezett proxy szerveren keresztül valósította meg, ami meggátolta a támadó kilétének az azonosítását.

## 1.4 APT-k

Az APT-k (Advanced Persistent Threat – magas szintű, folyamatos fenyegetést jelentő támadások) a 2010-es évek elején világosan bebizonyították, hogy képesek a hagyományos védelmi technológiák mellett is rendszerekbe behatolni és ott hosszú ideig észrevétlenül maradni, valamint gondoskodni értékes információknak a szervezeten kívülre történő küldéséről, azok eltulajdonításáról ([3]).

Az APT-k működésük során több különböző támadási lehetőség módszereit egyesítik, úgymint a social engineering módszereket a felhasználók átverésére ([2]), vírusterjedési módszereket újabb számítógépek felderítésére és megfertőzésére, illetve hálózati kommunikációs módszereket a kártékony kód távirányítására és az adatok kijuttatására.




*1. ábra: Egy tipikus célzott támadás és az APT (Advanced Persistent Threat) felépítése*

Az 1. ábra egy tipikus APT működését mutatja be, az alábbi legfontosabb részekre bontva:

1. **Intelligens felderítés (Intelligence Gathering):** Az első lépés során a támadó elsősorban nyilvános forrásból információkat gyűjt a leendő áldozatról, általában mint szervezetről, illetve az ott dolgozó munkatársakról. Nemcsak és kizárólag a szervezetre vonatkozó adatok érdeklik, hanem bármilyen személyes információ, amellyel akár egy alkalmazott bizalmába lehet férkőzni. A nyilvános források köre tehát nemcsak a szervezet weboldala, hanem például az alkalmazottak közösségi médiaoldalakon (például LinkedIn, Facebook) lévő profiloldalai. Az információk összegyűjtésével, a szervezet és alkalmazottainak a felderítésével előkészítheti az egyedi, testreszabott támadást. Például elég lehet, ha egy alkalmazottról kiderül a Facebook oldalán, hogy szereti a macskákat, máris megvan egy közös téma, amire hivatkozva kapcsolatot építhet ki vele a támadó.
2. **Bejutás (Point of Entry):** Egy támadás során a támadó a kártékony kódjának bejuttatására több módszer közül is választhat. A lehetőségeket azonban két lényeges csoportba oszthatjuk: egyrészt azokra a bejutási formákra, amelyek nem igényelnek felhasználói interaktivitást, illetve azokra, amelyeknek szükségük van a felhasználó beavatkozására. Az előbbi esetben a kiszemelt infrastruktúra valamely elemére vonatkozó sérülékenység kihasználása történik, mely általában az operációs rendszer, valamelyik alkalmazás vagy esetleg valamely alkalmazáson belül egy kiegészítő biztonsági problémához köthető. A másik csoportba azok a bejutási módszerek sorolhatók, amelyek során a felhasználónak például egy üzenetben érkezett csatolmányt meg kell nyitnia, vagy esetleg egy linkre kell kattintania. Az ilyen esetekben a felhasználót social engineering módszerekkel kell a támadónak rávennie, átvernie, hogy a kívánt cselekedetet végrehajtsa. Nyilvánvaló, hogy ez utóbbi esetben van különös jelentősége az intelligens felderítés során megszerzett információknak. Az 1.6 fejezetben részletesen foglalkozunk az ilyen social engineering támadásokkal.
3. **C&C kommunikáció (C&C – Command and Control Comunication):** Amennyiben a támadó sikeresen bejuttatta a kiszemelt környezetbe a kártékony kódját, a következő lépés során a támadó irányítása alá vonja a megtámadott eszközöket. Ennek érdekében egy saját C&C szervert állít fel, amelyen keresztül a megtámadott számítógépeket irányíthatja: parancsokat adhat, állományokat tölthet fel és le, lényegében bármit megtehet a megtámadott számítógépen, akár a háttérben is.
4. **„Oldalazó mozgás" (Lateral Movement):** A bejutást és a sikeres kapcsolatfelvételt követően a támadás következő lépésében a helyi hálózaton belüli további számítógépek és eszközök felderítése, majd az azokba történő behatolás következik. Ennek a célja, hogy további hozzáférési adatokat szerezzen meg, növelje a privilégium szintet, illetve, hogy biztosítsa a hálózat folyamatos felügyeletének a lehetőségét. A helyi hálózaton történő terjedést a támadó nyílván az 1. és 2. pont szerint akár kívülről is végrehajthatná, azonban egy infrastruktúra védelmi elemeinek jelentős része a külső kapcsolódási pontokon helyezkedik el, azaz egy belső számítógépről, eszközről történő behatolás egy másik számítógépre, eszközre sokkal egyszerűbb és a támadó szempontjából kevésbé kockázatos. Természetesen a támadó itt is bevethet social engineering trükköket, vagy akár biztonsági réseket is kihasználhat. A megtámadott számítógépek, eszközök irányítására a már használt C&C szervert is igénybe veheti.
5. **Értékes adatok, információk felderítése (Asset/Data Discovery):** A belső hálózaton történő terjedés során a támadó célja nem egyszerűen más számítógépek és eszközök felderítése, hanem ezeken az eszközökön tárolt értékes adatok és információk elérését biztosító szerverek és szolgáltatások azonosítása is. Ezt megteheti például a hálózati forgalom vizsgálatával.
6. **Adatok kiszivárogtatása (Data Exfiltration):** Miután az érzékeny adatokat, információkat a támadó azonosította, az adatokat általában először egy olyan belső számítógépre juttatja el, ahol egy elsődleges elemzés, válogatás és tömörítés történik, majd az így összeállított információk kijuttatása egy külső szerverre, amelyet a támadó közvetlenül elér.

# 1.5 Hálózati támadások

A US-CERT (United States Computer Emergency Readiness Team) szerint ([21]) **az incidens az a cselekedet, amelynek során megsértenek egy explicit vagy implicit biztonsági házirendet**, például:

- Sikeres vagy sikertelen kísérlet arra, hogy jogosulatlanul hozzáférést szerezzenek egy rendszerhez vagy annak adataihoz.
- Egy szolgáltatás nem kívánt megszakítása vagy megtagadása.
- Egy rendszer jogosulatlan használata adatok kezelése vagy tárolása céljából.
- A rendszer hardverének, firmware-ének vagy szoftverjellemzőinek a megváltoztatása a tulajdonos tudomása, utasítása vagy jóváhagyása nélkül.

Egy hálózati támadás első lépése szinte minden esetben a megcélzott számítógép operációs rendszerének és esetleg az azon futó alkalmazások megismerése. Ehhez kifinomult módszerek és eszközök állnak rendelkezésre. Ezt követhetik azok az eljárások és módszerek, amelyek már a konkrét támadást jelentik. Az alábbiakban a teljesség igénye nélkül ezeket a módszereket részletezzük.

### 1.5.1 Operációs rendszer megismerése, biztonsági rések kihasználása

Az operációs rendszer megismerése (OS fingerprinting) egy módszer annak meghatározására, hogy milyen operációs rendszer fut egy számítógépen. Gyakran ez a legelső lépés egy támadás során. Ha ugyanis egy támadó megtudja, milyen operációs rendszerrel fut a távoli, célba vett számítógép, akkor már könnyű feladat találni hozzá egy megfelelő támadó eszközt (exploitot).

Az operációs rendszer megismerésére a legegyszerűbb módszer a Telnet munkamenet indítása, ugyanis a sok Telnet szerver rengeteg információt ad az operációs rendszerről. A munkaállomások, szerverek telepítésekor, installálásakor azonban az egyik legelső lépés a Telnet szolgáltatás leállítása, hiszen ez a protokoll mindenfajta védelem nélküli kapcsolatot jelenthet a számítógéphez.

A Telnet híján az operációs rendszer és a hozzá tartozó környezet feltérképezésére, profilkészítésre a TCP használható. Az operációs rendszerek ugyanis különbözőképp valósítják meg a TCP-t, mely különbségek lehetőséget adnak az operációs rendszer azonosítására.

> A TCP (Transmission Control Protocol - [7]) egy kapcsolatorientált protokoll, azaz a két kommunikáló fél összeköttetést létesít, majd adatokat továbbíthatnak egymáshoz, végül lebontják a kapcsolatot. A TCP a kommunikációt full duplex (kétirányú) módon valósítja meg, azaz a küldés és fogadás egy időben történik. A kommunikáló feleknek nem kell törődniük a küldendő adatmennyiséggel, a hibakezeléssel; ezt mind megoldja a TCP.

A TCP-t használó módszer szerint speciális csomagokat küldenek a TCP szerverhez és figyelik a szerver válaszát. Az interneten számos eszköz szabadon elérhető, amely komplex módon megvalósítja ezt a stratégiát. Ilyen pl. az egyik legelterjedtebb eszköz, az Nmap ([6]).

Az operációs rendszer felderítésére a TCP például az alábbi módszerekkel használható:

- **FIN próba:** A TCP-ben a FIN csomag a kapcsolat lezárására szolgál, azaz létező kapcsolathoz kell tartoznia. A támadó csupán egy FIN csomagot küld a célszámítógép egy nyitott portjára, anélkül hogy kapcsolattal rendelkezne. A TCP-t definiáló RFC 793 szerint ([7]) erre nem kell válaszolni, de a Windows megvalósítások válaszolnak.
- **ACK érték:** A TCP-ben minden elküldött csomagot sorszámmal látnak el. A sorszámot a küldő fél folyamatosan, növekvő sorrendben generálja. A sorszámokat használják a megérkezett csomagok nyugtázására is (ACK érték). A csomagok sorszámozásához az operációs rendszerek azonban különböző tartományban lévő értékeket használnak.
- **ICMP hibaüzenetek:** A támadó a TCP segítségével olyan üzeneteket küld, amelyek a TCP eljárásában hibát eredményeznek. Ekkor a megcélzott szerver hibaüzeneteket küld, a kiküldött hibaüzenetek száma azonban korlátozott, és az operációs rendszertől függ.

A felsorolt módszereken túlmenően számos további lehetőség is kínálkozik a TCP-ben és a TCP-re vagy az UDP-re épülő további alkalmazási rétegbeli protokollokban az operációs rendszer felderítésére.

Az Nmap program az operációs rendszer felderítését több módszer szerint is elvégzi, majd az eredményeket összehasonlítja az egyes operációs rendszerekhez eltárolt sémákkal, így a tapasztalt viselkedéssel történő összevetéssel a legvalószínűbb operációs rendszer meghatározható.

Az operációs rendszer felderítését követően egy támadó már pontosan tudja, hogy az adott operációs rendszer milyen biztonsági hibákkal rendelkezik, rendelkezhet. Természetesen arra is következtethet, hogy melyek a leggyakoribb alkalmazások az adott operációs rendszer alatt, amelyeknek szintén lehetnek kihasználásra alkalmas biztonsági hibái. A biztonsági hibákat kihasználó eljárások (exploitok) egyik legnagyobb gyűjteménye a metasploit alkalmazás. A metasploit folyamatosan frissülő adatbázisa lehetőséget ad a legújabb sérülékenységek kihasználására is.

### 1.5.2 A kommunikációs protokollok támadása

Az interneten történő támadásoknak egyik elterjedt módja a szolgáltatás megtagadása (DoS – Denial of Service). Ezen módszerek használatával a támadó eléri, hogy a szerver túlterhelődjön, és ne legyen képes kiszolgálni a hozzá

beérkező további kéréseket. A módszer lényege, hogy nagyszámú feladat feldolgozása elé állítják a szervert, melynek a kiszolgálása időigényes, lényegesen több időt vesz igénybe, mint a "feladatot megfogalmazó" kérés előállítása és elküldése.

A teljesség igénye nélkül néhány internet protokoll támadására szolgáló módszer:

- **SYN Flooding:** A SYN Flooding (SYN elárasztás) támadások célpontja a TCP kapcsolat-felépítési eljárása. A kapcsolat felépítésének az első lépése a kapcsolat felépítésére vonatkozó kérés (SYN csomag) elküldése. A támadó nagyszámú SYN csomagot küld a távoli számítógépre és nem foglalkozik a válaszokkal. A SYN csomagok előállítása sokkal kisebb erőforrást (idő és tárhely) igényel, mint azok feldolgozása és kezelése, ezért a szerver könnyen túlterhelhető, előbb-utóbb nem lesz képes újabb kérések kiszolgálására.
- **Smurfing támadások:** Az ICMP (Internet Control Message Protocol) egy gyengeségét használja ki. Az ICMP protokoll segítségével „echo" (visszhang) csomagokkal ellenőrizhető, hogy vajon él-e, működik-e egy távoli állomás. Az „echo" csomagokban a küldő megadhatja, hogy milyen IP címre kéri a választ. A támadó hamisított IP címekkel készít „echo" csomagokat, ahol a hamisított IP cím az áldozaté, így a kérések elküldésével az áldozatot árasztják el a válaszok. Az ICMP szabványt 1999-ben módosították, hogy ellensúlyozzák az ilyen jellegű támadásokat.

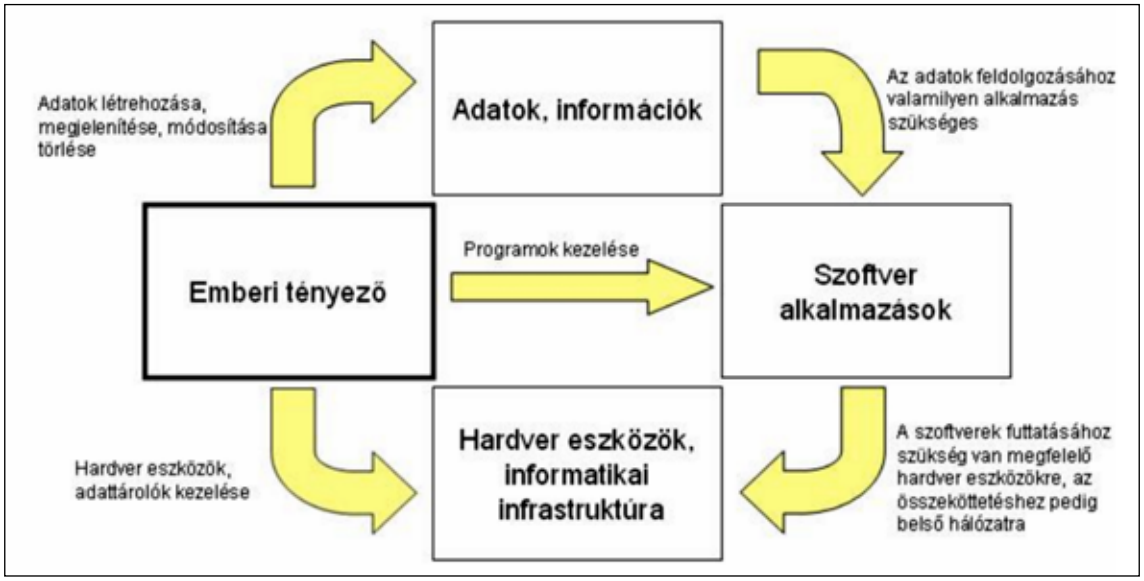
A szolgáltatásmegtagadásos támadásokat (DoS) a támadó a támadás elosztásával tovább fokozhatja. Az elosztott szolgáltatásmegtagadás támadásokat (DDoS – Distributed Denial of Service) a leggyilkosabb támadási formának tekinthetjük. A módszer lényege, hogy egy támadó nagyszámú gép felett veszi át az ellenőrzést és megszokott támadószoftvereket telepít rájuk. Egy adott jelre aztán a támadó szoftver üzenetekkel, csomagokkal kezdi bombázni a célba vett számítógép(ek)et.

# 1.6 Social engineering módszerek

A social engineering egy, az emberi tényező kihasználható tulajdonságaira építő támadási forma, tulajdonképpen olyan technikák gyűjteménye, amely az emberek befolyásolására, manipulálására alapozva teszi lehetővé bizalmas információk megszerzését, vagy éppen egy kártékony program terjedését és működését.([10]) Legmegfelelőbb definíciója a következő:

> *„A social engineering a befolyásolás és a rábeszélés eszközével megtéveszti az embereket, manipulálja, vagy meggyőzi őket, hogy a social engineer tényleg az, akinek mondja magát. Ennek eredményeként a social engineer – technológia használatával vagy anélkül – képes az embereket információszerzés érdekében kihasználni."*
>
> (Kevin D. Mitnick)

Az ilyen jellegű támadások veszélye tulajdonképpen abban rejlik, hogy ha belegondolunk, az emberi tényező az, ami bármihez hozzáfér, és valljuk be, amit a legkisebb erőfeszítésbe kerül átejteni. S hogy miért is a felhasználó, az ember a legkedvezőbb célpont egy támadó számára? Ahogyan az alábbi ábra is szemlélteti, az emberi tényező a legtöbb védendő értékhez közvetlenül hozzáférhet, ezáltal mint a biztonság leggyengébb láncszeme, vonzó célponttá válhat egy támadó szemében.




*2. ábra: Az emberi tényező hatása*

Általában az emberek számtalan olyan tulajdonsággal rendelkeznek, amelyeket egy támadó könnyen ki tud használni. Ezek közé a tulajdonságok közé tartozik a segítőkészség, amely az egyik legalapvetőbb emberi tulajdonság, amit a social engineerek számtalan módon ki tudnak használni, de hasonló kategóriát képez a kíváncsiság, hiszékenység, naivság, amelyre különösen adathalász támadások és kártékony programok beküldése során lehet építeni. A támadások során építeni lehet továbbá a felhasználók figyelmetlenségére, hanyagságára, illetve bizonyos értelemben tudatlanságára is.

Bár ezt az élet minden területén kihasználják, az SE kimondottan az információ megszerzésére irányul, ezen belül is elsősorban az informatikai eszközökön tárolt adatokra fókuszálva. Az évtizedek során felhalmozott tapasztalat szerint az IT eszközök védelme egyre kifinomultabbá vált, azonban az ezeket használó emberek biztonságtudatossága csak minimálisan növekedett. Így a legjárhatóbb támadási eljárás az emberi erőforrás kihasználása, hiszen ***a legtöbb biztonsági probléma a billentyűzet és a szék között található***. A támadónak több olyan emberi tulajdonságot is lehetősége van kihasználni, ami szinte kivétel nélkül minden potenciális áldozatban megtalálható. Az emberi hiszékenységgel való visszaélést számos tényező motiválhatja, így a social engineer-ek is több csoportba oszthatók. [18]

- Hackerek
- Ipari kémek
- Külföldi államok által megbízott hivatásos hírszerzők
- Személyes adatokat ellopásával foglalkozó bűnözők
- Elégedetlen munkavállalók
- Konkurens vállalkozások megfigyelői
- Magánnyomozók
- Csalók
- Fejvadászok (akár bűnügyi, akár munkajogi értelemben)
- Terroristák

Ebben a fejezetben a leggyakoribb, a felhasználók figyelmetlenségét, tudatlanságát kihasználó social engineering módszerek kerülnek bemutatásra.

### 1.6.1 Social engineering módszerek

A social engineering módszerek két nagy csoportba sorolhatók. Egyrészt beszélhetünk humán alapú módszerekről, amelyek közvetlen kontaktust feltételeznek a támadó és az áldozat között, másrészt azonosíthatunk számítógép alapú technikákat, amelyeknél a kapcsolat közvetett, a támadó valamilyen informatikai eszközön keresztül lép kapcsolatba az áldozattal. A humán módszerek a következők: [19]

- Segítség kérése
- Segítség nyújtása
- Kölcsönösség kihasználása
- Megszemélyesítés
- Shoulder surfing – képernyő lelesése
- Tailgating – bejutás a bejáraton más embert követve, annak tudtán kívül
- Piggybacking – bejutás a bejáraton más embert követve, annak tudtával
- Dumpster diving – információk felkutatása a hulladékban

A számítógép alapú támadások köre az alábbiak szerint alakul:

- Scam – hamisított weboldalak
- Adathalászat
- Phishing – E-mail alapú
- Vishing – VoIP alapú
- Smishing – SMS alapú
- Pharming – DNS eltérítésen alapuló
- Whailing – Vezetői IT eszközöket célzó támadás
- Baiting – Adathordozók szétszórása

### 1.6.2 Egy támadás felépítése

A social engineering jellegű támadások általában csak akkor hajthatóak végre sikeresen, ha a támadó megfelelően eltervezte az átverést. Ennek érdekében még a támadás előtt egy megfelelő forgatókönyvet kell készítenie. A szakiro-

dalom szerint az emberi tényezőt kihasználó támadások legtöbb esetben egy négy lépésből álló forgatókönyv szerint zajlanak [4]:

1. Információszerzés
2. Kapcsolat kiépítése
3. Kapcsolat kihasználása
4. Támadás végrehajtása

### 1.5.2.1 Információszerzés

Bármilyen támadásról legyen is szó, az első és legfontosabb lépés a megfelelő minőségű és mennyiségű információk összegyűjtése, különösen igaz ez a social engineering jellegű támadások esetében. Mivel ez a fázis meglehetősen alapos munkát igényel, különösen időigényes, az alkalmazott módszertől függően akár több hétig is eltarthat. Az információk gyűjtése – legyen szó akár valós támadásról, akár auditról - történhet az interneten keresztül, telefonos vagy személyes felkeresés során, levélben, vagy akár a célszemély vagy vállalat szemetének átvizsgálásával.

**Információk az internetről**

Manapság az információk gyűjtésének legegyszerűbb és legelterjedtebb módja az interneten való kutakodás. A keresgéléshez jó kiindulópontot jelentenek a vállalati honlap és a közösségi portálok, de a Google keresője is hasznos segítőtársnak bizonyul.

*Vállalati honlap*

A kiszemelt áldozat megismerésének legkézenfekvőbb módszere a cég honlapjának megtekintése. A vállalati weboldalra ugyanis gyakran kerülnek fel információk az alkalmazottakról, de legalábbis a vezetőségről, legtöbb esetben e-mail címekkel és telefonos elérhetőséggel. Néhány esetben – helytelen gyakorlatként – szervezeti ábra, belső telefonkönyv, sőt belső szabályzatok is megjelenhetnek nyilvánosan az oldalon. Ezekből az adatokból a támadó könnyen kiválaszthatja azokat az alkalmazottakat, akik a számára fontos információval rendelkezhetnek, illetve azokat a munkatársakat is, akiket egy esetleges támadás során, ha a szükség úgy kívánja, megszemélyesíthetnek, nem is beszélve a megszerzett e-mail címekről, amelyekre valamilyen fertőzött csatolmányt tartalmazó levelet küldhetnek.

*Közösségi portálok*

Napjainkban alig van ember, aki ne lenne regisztrálva legalább egy közösségi portálon. A nagyobb, népszerűbb közösségi oldalak (például Facebook) szintén remek kiindulópontot jelentenek egy social engineernek, hiszen innen gyűjthetik be a legtöbb személyes és kapcsolattartási információt a célszemélyekről. Nemcsak elérhetőségeket, személyes információkat és ismeretségeket lehet begyűjteni a szociális hálózatok segítségével, hanem a támadó akár jelszavak birtokába is juthat – amennyiben a felhasználó például valamelyik családtagja nevét, születési dátumát, házi kedvencének nevét használja jelszóként, és legyünk őszinték, sokan tesznek így.

A profiljukon nagyon sokan feltüntetik, mit csinálnak, amikor dolgoznak, így könnyen beazonosíthatóvá válik, mely területen is helyezkednek el, milyen forgatókönyvvel érdemes felkeresnie őket egy célzott támadás során a támadónak.

Szinte mindenki kitölti azt a rublikát is, hogy mit csinál szívesen szabadidejében – ezzel pedig megint csak könnyen egy célzottabb támadás áldozatává válhat, hiszen a támadó birtokába jut annak a tudásnak, hogy mi érdekli a célszemélyt, mivel tud hatni a kíváncsiságára, milyen tartalmú reklámlevelet nyitna meg feltételezhetően az áldozat. Tovább fokozza ezt a veszélyt az is, hogy több közösségi portál lehetőséget nyújt a kedvenc filmek, zenék, könyvek, stb. felsorolására is – ez egy sokkal kiélezettebb támadásra ad lehetőséget.

A legtöbb közösségi portálon lehetőségünk van különféle klubokhoz való csatlakozásra. Ez megint csak lehetőséget ad a támadónak, hogy megismerkedjen az áldozatul választott felhasználó érdeklődési körével, esetleg „klubtagként" felvegye vele a kapcsolatot.

A közösségi portálok ráadásul nemcsak az információk összegyűjtésére szolgálhatnak, hanem akár a kártékony programok, illetve azok forrásának terjesztésére is, hiszen a felhasználók – akár tudtukon kívül is – megoszthatnak kártékony kódot tartalmazó tartalmakat.

Érdekesség, hogy felmérések szerint azoknál a cégeknél, ahol az alkalmazottak rendszeresen látogatnak közösségi oldalakat, majdnem kétszer annyi biztonsági incidens történik, mint ott, ahol az ilyen jellegű oldalak látogatása nem engedélyezett. [8]

*Google*

Természetesen lehetőségünk van az interneten található információk keresőmotorokkal való felkutatására is. Ebben legnagyszerűbb segítőnknek a Google bizonyulhat, a Google operátorainak használatával ugyanis lehetőségünk van

nagyon specializált keresés indítására is – ezt a technikát nevezik Google Hacking-nek. A Google operátorainak, mint például intitle:, filetype:, inurl:, stb. használatával akár olyan, véletlenül a nyilvánosság előtt felejtett fájlok és oldalak kutathatók fel, amelyek hasznos információkat jelenthetnek egy támadónak. [9] Egy megfelelően felparaméterezett kereséssel akár egész címlisták is megtalálhatók nyilvánosan az interneten.

**Információszerzés telefonon keresztül**

Bár az interneten majdnem minden fellelhető, előfordulhat, hogy olyan információra van szüksége a támadónak, amelyet ott mégsem talál meg, vagy esetleg az előbbi módon beszerzett információk nem elegendőek a tervezett támadás kivitelezéséhez. Ekkor a legkézenfekvőbb megoldásnak egy telefonos megkeresés bizonyul. Az előzetesen összegyűjtött információk birtokában a támadó felhívhatja az ügyfél munkatársát, és a vállalat ügyfelének vagy partnercég alkalmazottjának, esetleg ha a cég méretei lehetővé teszik, akkor az adott cég egy másik részlegén dolgozó munkatársnak kiadva magát rákérdezhet a hiányzó információkra. Ilyen módon tájékozódhat tipikusan az adott területen illetékes kollégák kilétéről, elérhetőségükről, esetleg szabadságolásukról, de akár belső folyamatok, e-mail címlista, feladatok végrehajtásának megismerése is célkitűzés lehet.

**Információszerzés levélben, e-mail-en keresztül**

A levélben való megkeresés során kézenfekvő, ha a támadó a vállalat egy vagy több alkalmazottjával valamilyen kérdőívet töltet ki, amelynek kérdései közé gyanútlanul belecsempészi a számára fontos információkra irányuló kérdéseket. Ez lehet postai úton kiküldött levél, de sokkal gyorsabb és kényelmesebb az e-mailen keresztüli kommunikáció.

A kérdőívezés során a támadó elsősorban személyes adatokra, érdeklődési körre tehet szert, amelyeket későbbi célzott támadás során felhasználhat, illetve amelyek segítségére lehetnek az áldozattal való „közös hang" megtalálásában. De akár olyan eset is előfordulhat, hogy a social engineer egy célirányos levelet küld a cég alkalmazottainak például a munkahelyükkel való elégedettség mérésével kapcsolatban – így könnyen megtudhatjuk, kik azok a munkavállalók, akik elégedetlenek, s ezáltal nagyobb esély van rá, hogy áldozatává esnek egy támadásnak – például gondolkodás nélkül megnyitják a számukra véletlenül beküldött *berlista.xls* tartalmát.

Jogosan merülhet fel bennünk a kérdés, hogy hogyan lehet személyes adatokat megszerezni kérdőív kitöltetésekor, hiszen ezek általában név nélkül zajlanak, legtöbb esetben soha nem kötelezik az embert személyes adatok megadására. Mindenre van azonban megoldás: kisebb ajándékkal, nyereményjátékkal, személyes kapcsolatfelvétel ígéretével könnyen rá lehet venni a kitöltőt, hogy adja meg nevét, elérhetőségét is.

**Személyes felkeresés**

Bár Kevin Mitnick szerint az áldozat személyes felkeresése kell, hogy legyen a legutolsó, amit elkövet a támadó, hiszen a lebukás kockázata ekkor a legmagasabb, azonban néha előfordul, hogy a támadónak személyesen (is) fel kell keresnie a célszemélyt az információk megszerzése végett. [2] Az irodában való körbenézés során ugyanis számos, később felhasználható információt lehet találni, mint például szervezeti ábrát, naptárbejegyzéseket, szabadságolásokat, belső leveleket, szerződéseket, számlákat és egyéb dokumentumokat, nem is beszélve a monitorra ragasztott, jelszavakat tartalmazó cetlikről.

Mivel a személyes felkeresés tulajdonképpen egy megszemélyesítéses támadás, a támadó sokféle ember „bőrébe bújhat", lehet egy másik részleg munkatársa, ügyfél, partnercég alkalmazottja, futár, karbantartó vagy szerelő, hatósági személy, vagy egyszerűen egy, a szakdolgozatához segítséget kérő egyetemista is, sőt akár rendszergazda is. Azt, hogy a social engineer épp melyik szerepet választja, elsősorban az elérendő cél határozza meg. Ha a célja esetleg pusztán szakkifejezések elsajátítása vagy a belső folyamatok feltérképezése, választhatja az egyetemi hallgató szerepét, ellenben például bizalmas dokumentumokba való bepillantásért célszerűbb valamilyen külső auditor szerepében megjelennie, míg például egy billentyűzet-naplózó program feltelepítése egy felhasználó számítógépére inkább a rendszergazdaként való fellépés során lehet sikeres.

**Információk a szemetesből**

A social engineering módszerek közül a Dumpster Diving, vagyis a kuka-átvizsgálás tipikusan egy olyan tevékenység, amelynek célja belső információk keresgélése az irodai hulladékban. Bár a legtöbb vállalat rendelkezik iratmegsemmisítővel a leselejtezett bizalmas dokumentumok olvashatatlanná tétele végett, a tapasztalatok szerint mégis gyakori jelenség, hogy belső levelek, szerződések, jelenléti és szabadságolási ívek, pénzügyi jelentések, kinyomtatott e-mail címlisták és telefonkönyvek, illetve számtalan más, egy másik támadás során hasznosítható, bizalmasnak minősülő anyag végzi a kommunális hulladék között. Ha mégsem ez lenne a gyakorlat, akkor is gyakran tapasztalt eljárás, hogy a nagy mennyiségű megsemmisítendő iratokat tulajdonosa egyszerűen csak a berendezés mellett elhelyezett gyűjtő ládába helyezi el, mondván valaki majd ledarálja – megkönnyítve egyúttal egy lehetséges támadó dolgát is. Hasonló probléma merül fel a szelektív hulladékgyűjtéssel kapcsolatban is: a támadó így célirányosan tudja megközelíteni a tárolót, s még csak az ételmaradékok között sem kell turkálnia ...

### 1.6.2.2 Kapcsolat kiépítése

A megfelelő információk birtokában következhet a kihasználandó célszemély kiválasztása és a vele való kapcsolat kiépítése. A kapcsolatfelvétel történhet telefonon, e-mailben, közösségi portálon, stb. keresztül, vagy ritkább esetben akár személyesen is.

Legegyszerűbb, ha a támadó valamilyen segítséget kér a célszemélytől, vagy ennek fordítottja is megtörténhet, amikor ő segédkezik egy probléma megoldásában. Célravezető lehet valamilyen apró figyelmesség, kedvezményre feljogosítás, esetleg valamilyen „bizalmas" információ megosztása a kiszemelttel – mindennek pusztán a kapcsolat kiépítése, az esetleges gyanú eloszlatása a célja, sőt a támadó akár el is nyerheti az áldozata maximális bizalmát. Ha ezt sikerült elérnie, továbbléphet a következő fázisba, vagyis saját céljainak elérésére fordíthatja az újonnan kiépített kapcsolatot, és például beküldheti a korábbi levelezésben megígért – fertőzött – fájlt.

Kártékony programok esetében gyakori eset, hogy a kapcsolat kiépítése és a következő pontban bemutatott kihasználási fázis egy lépésben történik meg, de nem zárható ki célzott támadás esetén az sem, hogy a támadó a kártékony kód bejuttatása előtt egy korábbi levelezésben felveszi a kapcsolatot a célszeméllyel és megszerzi a bizalmát az általa küldött fájl biztos megnyitása érdekében.

### 1.6.2.3 A kapcsolat kihasználása

A bizalomépítés után előbb-utóbb elérkezik annak az ideje, amikor a támadó „szorul" segítségre, vagy kérhet szívességet, oszthat meg valami "bizalmas" információt, stb. Az esetek többségében a social engineer egyszerű szívességet kérhet áldozatától, például megkérheti egy bizalmas anyag kinyomtatására, segítséget kérhet belső bizalmas anyagok eléréséhez, de akár ráveheti arra is áldozatát, hogy töltsön le és futtasson le egy programot, sőt akár a saját felhasználónevével és jelszavával jelentkezzen be (esetleg adja meg azt) a számítógépes rendszerbe.

### 1.6.2.4 A tervezett támadás végrehajtása

Az eredetileg tervezett támadás bármi lehet, amit csak el tudunk képzelni, például a kártékony kód lefuttatásával a támadó távoli hozzáférést szerezhet az áldozat számítógépéhez, annak tudta nélkül.

Az eredetileg tervezett támadás akár a korábbi pontok során, azokkal egyidejűleg is megtörténhet. Ha azonban az előző pontban a támadó még nem érte volna el a célját, akkor a megszerzett információk és eszközök birtokában most lehetősége nyílik a károkozás végrehajtására. A cél bármi lehet, például a megszerzett felhasználónév-jelszó páros birtokában bejelentkezhet a rendszerbe, és módosíthat vagy törölhet fájlokat, esetleg a „kollégával" kinyomtattatott bizalmas dokumentumot nyilvánosságra hozhatja, vagy a kártékony kód lefuttatásával nyitott hátsóajtón keresztül távolról hozzáférhet a célszemély számítógépéhez. Mindezeknek nagy valószínűséggel valamilyen módon nem kívánt hatása lesz az adott vállalatra nézve (anyagi kár, jó hírnévvesztés, törvényi kötelezettségnek való elégtétel elmulasztása, stb.).

# 2. INFORMATIKAI RENDSZEREK SEBEZHETŐSÉGVIZSGÁLATA

Ebben a fejezetben az informatikai rendszerek biztonsági szintjének a vizsgálatára vonatkozó módszereket összegezzük a teljesség igénye nélkül. ([11]) A biztonsági tesztelések módszertanával több publikáció, ajánlás is foglalkozik, ezek közül a legjelentősebbek:

- az **Open Source Security Testing Methodology Manual**, amely az Institute for Security and Open Methodologies kiadványa [12],
- a **NIST SP 800-115 Technical Guide to Information Security Testing and Assessment** ajánlás, amely az amerikai kormányzat szabványa [13] és
- az **OWASP Testing Guide**, amit az Open Web Application Security Project keretében fejlesztenek [14].

Az informatikai rendszerek tesztelésének módszertanát ezek együttes használatával érdemes kialakítani. A biztonsági tesztelési módszertanok csoportosítására jelenleg nincs egyezményes megállapodás: a tesztelt rendszerek ismeretétől kezdve, a hozzáférés mértékén át, az alkalmazott tesztelési eszközökig számos taxonómia létezik.

## 2.1 Tesztelési eljárások

Az informatikai rendszerek fejlesztése során, azok tesztelésén keresztül egészen a működő rendszer üzem közbeni vizsgálatáig számos eljárás létezik, amelyek kisebb-nagyobb mértékben a rendszer biztonsági kérdéseit is érintik. Egy informatikai rendszerrel szemben támasztott biztonsági követelmények kialakítása és ennek ellenőrzése már az informatikai rendszer tervezésekor és megvalósításakor elkezdődik. A tervezés során, a rendszer specifikációjának a kialakítását az alábbi lépésekben végezhetjük el:

1. **Követelmények specifikálása:** A szoftverek fejlesztési folyamatainak kiindulópontja minden esetben az informatikai rendszerre vonatkozó követelmények specifikálása. Ennek során rögzíteni kell az informatikai rendszerre vonatkozó funkcionális követelményeket, ami azt tartalmazza, hogy mely funkciókat és milyen módon kell a rendszernek megvalósítania. A fejlesztési fázis eredményeként létrejön a *funkcionális követelmények dokumentációja*.
2. **Kockázatanalízis:** Ennek a fázisnak a célja, hogy az informatikai rendszerben a lehetséges veszélyhelyzeteket meghatározzuk. Itt olyan veszélyhelyzeteket kell feltárni, amelyek valóságos vagy lehetséges veszélyforrást jelentenek a személy-, a környezet-, illetve a vagyonbiztonság szempontjából. Ezt követően pedig a feltárt veszélyhelyzetekhez valószínűségi alapon a kockázatot is meg kell határozni. A kockázatanalízis eredményeit a *biztonsági követelmények dokumentációjában* kell rögzíteni. Nyilvánvaló, hogy a rendszer által tárolt és kezelt adatokra, illetve az egész rendszerre vonatkozóan ekkor kerülnek rögzítésre az informatikai biztonsággal kapcsolatos elvárások is.
3. **Rendszer-specifikálás:** A funkcionális követelmények, illetve a biztonsági követelmények összevetésével a két dokumentum együttesen alkotja a rendszer-specifikációt. Az összevetés eredményeként a rendszer-specifikáció a funkcionális követelmények dokumentációjához képest már a biztonsági szempontokat is figyelembe veszi.

Egy informatikai rendszer fejlesztése során és az üzemelés alatt az alábbi típusú teszteléseket végezhetjük el:

- A **funkcionális tesztelés** során a programnak kizárólag a külső viselkedését, a specifikációban részletezett funkcióinak teljesítését vizsgáljuk. A tesztelést a bemeneti adatokra adott kimeneti adatok megfigyelésével végezzük el, amihez a futtatható programot használjuk. A funkcionális tesztelés során nem áll rendelkezésre a program forráskódja, kizárólag a futtatható tárgykód, amit felhasználhatunk. Ezt a megoldást ezért *fekete doboz* (*black-box testing) tesztelésnek* is szokták nevezni. A funkcionális tesztelés során a tesztelésnek az a célja, hogy egyrészt vegye sorba és ellenőrizze a program minden egyes funkcióját, mely a specifikációban részletezett, másrészt, hogy végezzen vizsgálatokat arra vonatkozóan, hogy a programnak nincs-e valamilyen hibás tevékenysége.
- A **strukturális tesztelés** során a teszttervezés, a tesztesetek definiálása a szoftver forráskódjának felhasználásával, a belső struktúra, a belső működés ismertében történik. Ebben a megközelítésben a tesztesetek megtervezésekor az a cél, hogy a forráskód utasításainak és elágazásainak minél alaposabb végigjárását tudjuk biztosítani a tesztesetek összességében. Itt a szoftvert, annak minden belső tulajdonságával együtt egy ismert egységnek tekintjük, emiatt szokás ezt a módszert *fehér doboz tesztelésnek (white-box testing)* nevezni. A strukturális tesztelés tehát a program vezérlési szerkezetének, annak minden egyes utasítása végrehajtásá-

nak, a döntési elágazások lehetséges irányba történő végrehajtásának, valamint a ciklusok működésének az ellenőrzésére irányul.

Mind a funkcionális, mind pedig a strukturális tesztelés a biztonsági szint ellenőrzésére is ki kell, hogy terjedjen, azonban az ilyen általános célú tesztek nem elsősorban a biztonsági kérdésekre irányulnak. Az alábbi néhány, a szakirodalomban is előforduló vizsgálati eljárás azonban szorosan kapcsolódik az informatikai rendszerek biztonsági vizsgálatához, teszteléséhez:

- **Sérülékenységvizsgálat:** a rendszer (alkalmazás) védelmi mechanizmusában előforduló sebezhetőségek felderítése, tipikusan automatikus, de akár manuális eszközökkel. Ezek a vizsgálatok alapvetően fekete-doboz tesztelésként használatosak, a belső struktúra ismerete nélküli, modulszintű vizsgálatnak tekinthetőek. Például egyes bemeneteken olyan adatok megadása, amelyekre a rendszer egyébként védett információkat ad át. Összefoglalva alkalmazásszintű ellenőrzésnek nevezhetjük ezt az eljárást.
- **Behatolás-tesztelés:** az újonnan fejlesztett vagy már működő rendszer védelmi kontrolljainak kijátszása műszaki megoldásokkal, bármilyen kapcsolódó környezeti infrastruktúra felhasználásával. Két alfaja létezik, a Blue teaming, amelynek során a tesztelő pontosan ismeri a tesztelt infrastruktúrát, és a Red teaming, amelynél semmilyen ismerettel nem rendelkezik. Ezt a fajta tesztelést rendszerszintű ellenőrzésnek kell felfogni.
- **Etikus hackelés:** a működő rendszer védelmi mechanizmusának kijátszása bármilyen technikával, a rendszer előzetes ismerete nélkül. Ebbe beletartozik az emberi ráhatással (social engineering) történő támadás is Ez a **szervezet szintű ellenőrzés** definíciója.

## 2.2 Támadói profilok a Common Criteria alapján

Egy alkalmazás biztonsági teszteléséhez a legjobb kiindulópontot a Common Criteria (CC) szabvány adja, amelynek Common Evaluation Methodology (CEM) című kiadványának B melléklete részletesen leírja egy értékelő feladatait. [15] A Common Criteria terminológiájában **a sebezhetőségi felmérés az a folyamat, melynek során a termék hibáinak vagy gyengeségeinek a létét és kihasználhatóságát elemzik**. A tesztelést az értékelő végzi értékelői tesztelés módszerével. Alapszinten csak a nyilvános forrásból hozzáférhető sebezhetőségeket kell felderíteni (az ISO 27002 és a KIB 28. ajánlásoknak megfelelően), ám a rendszer biztonsági besorolása szerint ennél sokkal részletesebb vizsgálatokra is sor kerülhet. A sérülékenységvizsgálat három lépésből áll:

- a lehetséges sebezhetőségek felderítése,
- a sebezhetőség kockázati besorolása és
- a sebezhetőség kihasználása annak megerősítésére, hogy a hiba az adott környezetben valóban kihasználható.

A Common Criteria a sebezhetőségeket öt típusba sorolja:

- **Megkerülés (bypassing):** Ebbe a kategóriába tartozik minden olyan hiba, amelynek során a rendszer beépített biztonsági eljárásait megkerülik, pl. jogosulatlan hozzáférés, kriptográfiai védelem megkerülése.
- **Meghamisítás (tampering):** Olyan sebezhetőségek, amelynek kihasználásával a termék működése módosítható, pl. a biztonsági funkciók leállítása vagy a fizikai módosítás.
- **Direkt támadások (direct attacks):** A permutációt vagy véletlenszerűséget felhasználó védelmi eljárások tesztelése tartozik ebbe a körbe, pl. jelszóhosszúságokon alapuló támadások.
- **Megfigyelés (monitoring):** Az információk átvitelének vagy kiszivárgásának figyelése és felhasználása, pl. információs csatornák lehallgatása, rejtett csatornák alkalmazása, elektromágneses sugárzás figyelése.
- **Nem megfelelő használat (Misuse):** Ez alatt a termék nem megfelelő dokumentációját, helytelen konfigurációját és szokásostól eltérő használatát értjük.

A fenti hibák felderítésére az értékelőnek két lehetséges módja létezik. Egyrészt a vizsgálat során szembesülhet olyan sebezhetőségekkel, amelyek kihasználása nem triviális, de minden valószínűség szerint lehetséges. Ezek többnyire olyan véletlen felfedezések, amelyek a tesztelés „melléktermékei", nem direkt ezekre volt kíváncsi a tesztelő. Példa lehet erre egy olyan teszteredmény, amely a termékben puffertúlcsordulást okoz, ami akár a védelmi funkciók megkerülését is lehetővé teheti alaposabb elemzés után. A másik lehetséges elemzési mód valamilyen elvek alapján felépített analízis. Ez lehet

- nem strukturált, amelynek során az értékelő saját tapasztalata alapján általános sebezhetőségeket keres,
- fókuszált, ami egyes sebezhetőnek tűnő területek alapos elemzését jelenti, vagy
- módszertan alapján történő, amely a korábban felsorolt ajánlásokra épül.

Az értékelési eljárás azonban elsősorban a támadási potenciál szintjétől függ, aminek meghatározásához a Common Criteria kiváló segédletet nyújt: a vizsgálati körülmények alábbi öt paramétere segítségével meghatározható, hogy a tesztelés milyen támadási potenciált képvisel:

- **a sebezhetőség sikeres kihasználáshoz szükséges idő (eltelt idő)**, lehetséges értékei: egy napnál kevesebb, egy nap és egy hét között, egy hét és két hét között, két hét és egy hónap között, egy hónap és hat hónap között, hat hónapnál több;
- **a kihasználáshoz szükséges szakértelem (szakértelem szintje)**, lehetséges értékei: laikus (általános ismeretei vannak), profi (jól ismeri az adott terméktípust), szakértő (széleskörű ismeretei vannak a támadási technikákról), több szakértő (részterületek szakértői);
- **a vizsgált termék felépítésének és működésének ismerete (termékismeret)**, lehetséges értékei: nyilvános információk (internetről elérhető), korlátozott információk (a fejlesztői közösségen belül ismert), érzékeny információk (csak egy speciális fejlesztői csapat által ismert), kritikus információk (csak néhány személy által ismert);
- **a sikeres támadáshoz szükséges hozzáférés ideje és a próbálkozások száma (próbálkozási ablak),** lehetséges értékei: korlátlan (a támadás észrevétlen marad), egyszerű (egy napnál rövidebb ideig tartó hozzáférés, 10-nél kevesebb próbálkozás), mérsékelt (egy hónapnál rövidebb ideig tartó hozzáférés, 100-nál kevesebb próbálkozás), nehéz (legalább egy hónapig tartó hozzáférés vagy legalább 100 próbálkozás), nincs (a rendelkezésre álló idő vagy próbálkozásszám nem elégséges egy sikeres támadás véghezviteléhez).
- **a támadás kivitelezéséhez szükséges hardverek és szoftverek (eszköztár)**, lehetséges értékei: szabványos eszközök (internetről szabadon letölthető), speciális eszközök (piacon beszerezhető), egyedi eszközök (adott célra fejlesztett), több egyedi eszköz (résztámadások egyedi eszközei).

A Common Criteria minden lehetséges értékhez egy számot rendel, melyek összege adja a támadási potenciált. A forgatókönyvek összeállításánál a biztonsági szinttől függően külső vagy belső támadókkal számolhatunk. Az informatikai rendszerek üzemeltetői általában azzal számolnak, hogy a legitim felhasználóik megbízhatóak, ezért a védelmet sokszor eszerint építik fel. Természetesen a gyakorlat azt mutatja, hogy az ilyen támadásokat is szimulálni kell.

# 2.3 Tesztelési módszerek

## 2.3.1 Forráskód- és alkalmazásszintű vizsgálatok

A Common Criteria a már említettek szerint két oldalról közelíti meg a biztonságtesztelés kérdését. Ezek mindig az alkalmazásra koncentrálnak, nem foglalkoznak az informatikai környezetet alkotó rendszerelemek biztonságával. A fejlesztőnek egyrészt bizonyítania kell az alkalmazásban implementált biztonsági funkcionalitás működőképességét, másrészt egy független értékelőnek körültekintő sebezhetőségvizsgálatot kell végrehajtania.

A biztonsági funkcionális tesztelésre a Common Criteria ATE osztálya szab előírásokat, amelyek elsősorban a fejlesztőkre vonatkoznak:

- ATE_COV.2: az implementált biztonsági funkciók tesztlefedettségének ellenőrzése. Eszerint minden biztonsági funkció interfészét le kell ellenőrizni, és ezt a tervezési dokumentációval összhangban kell megtenni.
- ATE_DPT.1: a tesztelési mélység elemzése. Ennek során azt kell bizonyítani, hogy minden egyes biztonsági funkció le lett tesztelve.
- ATE_FUN.1: a funkcionális tesztelés formai követelményeit határozza meg.
- ATE_IND.2: a független biztonsági funkcionális tesztelés módszerének meghatározása. Eszerint a független értékelőnek a fejlesztők által elvégzett biztonsági funkcionális tesztelés bizonyos részeit mintavételezett eljárással újra el kell végeznie.

A sebezhetőségvizsgálatot független értékelőnek kell végrehajtania a Common Criteria AVA osztályának útmutatása szerint:

- AVA_VAN.2: Az értékelőnek ebben az esetben először nyílt forrásokban kell ellenőriznie, hogy az alkalmazásban van-e bármilyen lehetséges sebezhetőség. Ezután az útmutató dokumentumok, a funkcionális specifikáció, a tervezési dokumentumok és a biztonsági architektúra-leírás alapján kell sebezhetőségeket keresnie, ez azonban nem terjed ki a forráskód ellenőrzésére. Végül az azonosított sebezhetőségek mentén ki kell derítenie, hogy az adott hiba kihasználható-e. Tipikus alkalmazásszintű vizsgálatról van tehát szó.
- AVA_VAN.3: az AVA_VAN.2-höz képest annyi eltérést tartalmaz, hogy kódszintű ellenőrzést is végre kell hajtani.

Az OWASP Application Security Verification Standard Project (ASVS) a Common Criteria-hoz hasonló elvek alapján négy, különböző szintű ellenőrzésre tesz javaslatot az elkészült alkalmazásban, így ezek hasznos kiegészítői lehetnek a Common Criteria szabványnak. [16] Az ASVS négy szinten határozza meg a webes alkalmazásokra vonatkozó biztonsági ellenőrzések körét, kimondottan a biztonsági funkcionális tesztelés megközelítéssel, azaz az infrastruktúra-elemek nincsenek a célkeresztben, csak az alkalmazás. Bár a dokumentum hivatalosan webes alkalmazásokra

használható, kellően magas szintű ahhoz, hogy analóg módon más típusú alkalmazásokat is fel lehessen vele mérni. A Common Criteria-hoz képest jelentős előnye, hogy a biztonsági funkciókra konkrétabb tesztelési követelményeket határoz meg, mint a Common Evaluation Methodology.

Az ASVS négy szintje a következő:

- Az eszközökkel végrehajtott vizsgálat (Level 1) célja meggyőződni arról, hogy semmilyen kártékony kód nem került az alkalmazásba. Ez történhet a bináris kód dinamikus ellenőrzésével vagy a forráskód statikus ellenőrzésével. Ideális esetben mindkét ellenőrzést le kell futtatni. Ez a vizsgálat csak közvetlenül a fejlesztett kódot érinti.
- A manuális tesztelés és átvizsgálás (Level 2) célja a fejlesztett kódok, illetve a biztonsági funkcionalitást biztosító, harmadik féltől származó komponensek vizsgálata. A vizsgálat során az eszközök használata támogatott, de a kívánt bizonyosságot nem kizárólag az eszköz által szolgáltatott eredmény adja, a tesztelőnek manuális módszerekkel is alá kell támasztania a megállapításokat. Ez a vizsgálat szintén két altesztelési módszerből tevődik össze. Egyrészt manuális behatolási tesztelést kell végrehajtani, valamint manuális kódelemzés is elvárt.
- A manuális átvizsgálás a tervek ismeretében (Level 3) a Level 2 szinthez képest annyiban tér el, hogy meg kell vizsgálni a harmadik féltől származó komponensek kódjait is.
- A manuális átvizsgálás a tervek és a kód ismeretében (Level 4) tulajdonképpen teljes belső átvizsgálást jelent, amelynek hatóköre kiterjed a fejlesztőeszközökre, a fordítókra, mindenre, aminek köze volt a kód létrehozásához.

Az ASVS az alábbi területeken javasol vizsgálatokat:

- Biztonsági architektúra
- Hitelesítés
- Munkamenet-kezelés (session)
- Hozzáférés-vezérlés
- Input validáció
- Kimenet kódolása, escape-elés
- Kriptográfia
- Hibakezelés és naplózás
- Adatvédelem
- Kommunikáció biztonsága
- HTTP biztonság
- Biztonsági konfiguráció
- Kártékony kód keresése
- Belső biztonság

### 2.3.2 Rendszerszintű vizsgálatok

Az Open Web Applications Security Project (OWASP) keretében 2003 óta rendszeresen közzéteszik a webes alkalmazásokra vonatkozó nagy kockázatú sebezhetőségeket. Az OWASP Top 10 alapján a sérülékenységvizsgálat olyan módszertana állítható össze, amely segít a legtipikusabb hibák felderítésében, így jelentősen növekedhet az alkalmazás biztonsági szintje. A rendszert érintő vizsgálatok során elsősorban ezekre célszerű koncentrálni, függetlenül attól, hogy épp az alkalmazást vagy valamelyik azt futtató infrastruktúra-elemet vizsgáljuk. A tesztesetek webes környezetre vannak kidolgozva, de analóg módon a legtöbb esetben más platformokon is használhatók. A legnagyobb kockázatú hibák az alábbiak:

- **A1 – Beszúrásos támadások (Injection):** A beszúrásos hibák, amelyek közé elsősorban az SQL, operációs rendszer és LDAP injection értendő, olyankor történnek, amikor a támadó nem megbízható adatokat küld a parancsfeldolgozó felé parancsként vagy lekérdezésként. A támadó kód eléri a parancsfeldolgozónál, hogy az nem kívánt parancsot hajtson végre, vagy érzékeny adatokat szivárogtasson ki.
- **A2 – Cross-Site Scripting (XSS):** Az XSS hibák jellemzője, hogy egy alkalmazás nem megbízható adatokat vesz át és küld tovább a böngészőn keresztül megfelelő ellenőrzés és szűrés nélkül. Az XSS lehetővé teszi a támadónak, hogy szkripteket hajtson végre az áldozat böngészőjében, amivel el tudja téríteni a felhasználó session-jét, weboldalakat tud megváltoztatni, vagy át tudja irányítani a felhasználót egy kártékony oldalra.
- **A3 – Hibás hitelesítés és sessionkezelés (Broken Authentication and Session Management):** Az alkalmazások hitelesítéshez és sessionkezeléshez kapcsolódó funkciói sok esetben nem megfelelően lettek implementálva, ezért lehetővé válik a jelszavak, kulcsok, session tokenek megszerzése, vagy más hibák előidézése, aminek segítségével a támadó más, jogosult felhasználó nevében tud eljárni.

- **A4 – Nem biztonságos direkt objektumhivatkozás (Insecure Direct Object References**): Direkt objektumhivatkozásról akkor beszélünk, amikor a fejlesztő felfed egy hivatkozást valamilyen belső, implementációhoz szükséges objektum felé, mint pl. egy fájl, könyvtár vagy adatbázis tábla. Megfelelő hozzáférés-védelem vagy más biztonsági megoldás nélkül a támadó vissza tud élni ezekkel a hivatkozásokkal, és nem jogosult hozzáférést szerezhet az ezekben tárolt adatokhoz.
- **A5 – Cross-Site Request Forgery (CSRF):** A CSRF támadás során a támadó kényszeríti az authentikált felhasználó böngészőjét arra, hogy egy hamisított HTTP kérést küldjön egy sebezhető webalkalmazás felé, amely tartalmazza az áldozat session cookie-ját és más hitelesítési információkat. Ez lehetővé teszi, hogy az áldozat böngészője olyan kéréseket küldjön a sebezhető alkalmazás felé a támadó nevében, melyről azt hiszi, hogy az legitim forrásból érkezik.
- **A6 – Helytelen biztonsági beállítások (Security Misconfiguration):** A megfelelő biztonság eléréséhez meg kell határozni az alapvető beállításokat, és ezeket meg is kell valósítani az alkalmazásokban, keretrendszerekben, alkalmazásszervereken, webszervereken, adatbázisszervereken és minden más érintett platformon. Mivel a legtöbb rendszer nem olyan alapbeállítással kerül telepítésre, amely az elvárható biztonsági szintet valósítja meg, ezeket a konfigurációkat meg kell határozni, implementálni kell, és folyamatosan fenn kell tartani. A folyamat során a szoftverek frissítéseire is figyelemmel kell lenni.
- **A7 – Nem megfelelő kriptográfiai tárolás (Insecure Cryptographic Storage):** Számos webalkalmazás nem megfelelően kezeli az érzékeny adatokat, mint például a hitelesítési adatok, mert ezeket nem titkosított vagy lenyomatolt formában őrzi. A támadók ezért megszerezhetik vagy módosíthatják a gyengén őrzött adatokat, ami számos visszaéléshez vezethet.
- **A8 – URL hozzáférés korlátozásának hibája (Failure to Restrict URL Access):** A webes alkalmazásokban fontos az URL-ek hozzáférési jogainak ellenőrzése, mielőtt a felhasználó elérne egy védett hivatkozást vagy akciógombot. A hozzáférési jognak ezt az ellenőrzését viszont minden esetben meg kell tenni, amikor ezeket a védett oldalakat elérik, mert különben a támadó elérheti a rejtett oldalakat.
- **A9 – Nem megfelelő szállítási réteg védelem (Insufficient Transport Layer Protection):** Az alkalmazások gyakran nem megfelelő hitelesítést, titkosítást és bizalmasság-sértetlenség védelmet használnak az érzékeny hálózati forgalomban. Amikor viszont használnak, akkor is gyenge algoritmusokkal, lejárt vagy érvénytelen tanúsítványokkal teszik ezt, vagy egyszerűen nem megfelelően használják a kriptográfia adta lehetőségeket.
- **A10 – Nem ellenőrzött átirányítások és továbbítások (Unvalidated Redirects and Forwards):** A webes alkalmazások gyakran irányítják át vagy továbbítják a felhasználókat más oldalakra, és használnak nem hiteles adatokat a forrásoldal megállapítására. Megfelelő ellenőrzés nélkül a támadók átirányíthatják a felhasználókat adathalász vagy kártékony oldalakra, vagy a továbbításokkal nem jogosult hozzáférést szerezhetnek.

A fenti hibák Common Criteria szerinti besorolása a következő:

- Megkerülés: A2, A4, A5, A8
- Meghamisítás: A1, A7, A10
- Direkt támadások: A3, A7
- Megfigyelés: A9
- Nem megfelelő használat: A6

Behatolás-tesztelés és etikus hackelés esetén szükséges az alkalmazáson kívül akár a teljes informatikai infrastruktúra automatizált vizsgálata is. A teljesség kedvéért meg kell említeni az infrastruktúra ellenőrzésére használt automatizált eszközök körét is. Ezek közé tartoznak azok a szoftverek, amelyekkel az infrastruktúra felderítését lehet végrehajtani, amelyekkel az operációs rendszerek válnak támadhatóvá, amik a hálózati elemekkel szemben hatékonyak, végül azok, melyek az alkalmazás közvetlen környezetét jelentő szoftvereket (adatbázisok, alkalmazás-szerverek) vizsgálják. Ezek együttes, célravezető használata támogatja a teljes körű vizsgálatot. [17]

## 2.4 Sebezhetőség-vizsgálati módszertan

A webes alkalmazások teszteléséhez az Open Web Application Security Project keretében megalkotott módszertanok tekinthetők a leghasznosabbaknak, legalábbis ezekre hivatkozik a legtöbbet a szakirodalom napjainkban. Ezek a módszerek azonban nemcsak webes környezetben használhatók. Az OWASP két csoportba sorolta az ellenőrzési folyamatokat:

- Az OWASP Code Review Guide (CRG) célja a létrehozott kódok manuális ellenőrzése, white-box módszerrel. [20]
- Az OWASP Testing Guide (TG) a már elkészült alkalmazás black-box teszteléséhez, azaz sérülékenységvizsgálatához ad segítséget.

A CRG tehát a kódok átvizsgálására készült, amely folyamat célja meggyőződni arról, hogy a forráskódban a megfelelő biztonsági kontrollok implementálásra kerültek és úgy működnek, ahogy a tervezés szerint működniük kell. Ellenőrzi továbbá azt is, hogy a biztonsági kódokért felelős fejlesztők követték a számukra megállapított eljárásrendeket. Mivel a behatolástesztelés, de még a sérülékenységvizsgálat sem képes teljes körűen felderíteni az esetleges hibákat, egyedül a white-box tesztelésnél van lehetőség a rosszul megírt kódrészleteket átfogóan megtalálni. Ez a tesztelés a humán erőforrás és a célszoftverek alkalmazását is szükségessé teszi. Az automatizált eszközök itt a leghatékonyabbak, de nem képesek teljes mértékben pótolni a tesztelési szakembert, hiszen az egyes kontextusok döntően befolyásolhatják a kódolás megfelelőségét vagy nem megfelelőségét.

A CRG a legfontosabb technikai kontrolloknak az authentikációt, az authorizációt, a sessionkezelést, az input validációt, a hibakezelést, az alkalmazás telepítését és a kriptográfiát tartja, többé-kevésbé összhangban a Common Criteria biztonsági funkcionális követelményeivel. A legkomolyabb programozáskor elkövetett hibaforrásoknak pedig a puffertúlcsordulást, az operációs rendszer és SQL injektálást, az adatvalidációt, a cross-site scriptinget, a cross-site request forgery-t, a naplózási hiányosságokat, a session sértetlenségét és a versenyhelyzeteket tartja, szinkronban az OWASP Top 10 megállapításaival. A CRG részletesen bemutatja azokat a tipikus hibákat, amiket a biztonsági funkciók implementálásánál el szoktak követni, illetve azt is, hogyan lehet a programozási hibákat felderíteni.

A CRG-ben meghatározott tesztelés azonban idő- és erőforrás-igényes. Költséghatékonyabb megoldás a TG-ben leírt sérülékenység-vizsgálatokat végrehajtani, ami ugyan nem jelent ugyanakkora bizonyságot, mint a CRG folyamatai, de így is jelentősen tudja csökkenteni a támadási felületeket. A TG az OWASP Top 10 fenyegetéseit veszi sorra, és leírja, hogy milyen tesztelési eljárásokkal lehet meggyőződni arról, hogy a kész alkalmazás ellenáll ezeknek. Black-box tesztelés esetén az automatikus teszteszközök már körülményesebben használhatók, de még bevethetők, ezért az egyes tesztesetekhez konfigurálható céleszközöket szoktak használni, ami feltételezi, hogy a tesztelő jól ismeri ezt a területet. A TG 10 kategóriában 66 tesztesetet határoz meg, amelyeket az alkalmazás típusától és megvalósításától függően célszerű végrehajtani:

1. Információszerzés:
   - Webes keresők robotjai
   - Felderítés webes keresők adatbázisában
   - Alkalmazás belépési pontjainak azonosítása
   - A webes alkalmazás ujjlenyomatának tesztelése
   - Alkalmazás felderítése
   - Hibaüzenetek elemzése
2. Konfigurációmenedzsment tesztelés
   - SSL/TLS tesztelés
   - DB Listener tesztelés
   - Infrastruktúra konfigurációmenedzsmentjének tesztelése
   - Alkalmazás konfigurációmenedzsmentjének tesztelése
   - Fájlkiterjesztés kezelésének tesztelése
   - Régi, mentett és nem hivatkozott fájlok
   - Infrastruktúra és alkalmazás adminisztrátori interfészek
   - HTTP metódusok és XST tesztelés
3. Authentikációs eljárások tesztelése
   - Authentikációs adatok átvitele biztonságos csatornán keresztül
   - Felhasználói adatbázis tesztelése
   - Kitalálható (szótár alapú) felhasználói fiókok felderítése
   - Nyers erejű tesztelés
   - Az authentikációs séma kikerülésének tesztelése
   - Sebezhető emlékeztető jelszó és jelszó visszaállítási lehetőségek
   - Kilépés és böngésző cache menedzsment
   - CAPTCHA kódok
   - Többfaktorú authentikáció tesztelése
   - Versenyhelyzetek kezelése
4. Sessionkezelés
   - A sessionkezelési séma tesztelése
   - Cookie attribútumok
   - Session fixálás
   - Kiterjesztett sessionváltozók tesztelése
   - CSRF

5. Authorizáció tesztelése
   - Elérési útvonalak tesztelése
   - Az authorizációs séma kikerülése
   - Jogosultság kiterjesztése
6. Üzleti logika
   - Az üzleti logika tesztelése
7. Adatvalidáció:
   - Nem perzisztens XSS
   - Perzisztens XSS
   - DOM alapú XSS
   - Cross Site Flashing
   - SQL Injection
   - LDAP Injection
   - ORM Injection
   - XML Injection
   - SSI Injection
   - XPath Injection
   - IMAP/SMTP Injection
   - Kódbeszúrás
   - Operációs rendszer parancsainak kiadása
   - Puffer túlcsordulás
   - Összetett adatvalidációs tesztelés
   - HTTP splitting/smuggling tesztelés
8. Túlterheléses támadások (DoS) tesztelése
   - SQL wildcard támadás
   - Ügyfélfiókok zárolása
   - DoS puffer túlcsordulások tesztelése
   - Felhasználó által meghatározott objektumallokáció
   - Felhasználói adatbevitel, mint körbeforgó számláló
   - Felhasználó által bevitt adatok lemezre írása
   - Erőforrás sikertelen elengedése
   - Túl sok adat tárolása egy session-ben
9. Web service tesztelés
   - Web service információszerzés
   - WSDL tesztelés
   - XML struktúra tesztelés
   - XML tartalom szintű tesztelése
   - HTTP GET paraméterek/REST tesztelés
   - SOAP csatolások
   - Visszajátszásos támadás
10. AJAX tesztelés
    - AJAX sebezhetőségek
    - AJAX tesztelés

# 3. SOCIAL ENGINEERING VIZSGÁLATI MÓDSZEREK

A social engineering alapú biztonsági tesztelést végrehajtó a lehetséges SE technikák közül bármelyiket választhatja, ami a kívánt cél elérésében számára a legnagyobb segítséget nyújtja. A szervezeti biztonságot az összes adminisztratív, fizikai és logikai védelmi kontroll együttes használata jelenti, a social engineering ezek mindegyikét próbára teszi, ezért lehet ezt a módszert a szervezeti biztonság sebezhetőségvizsgálatának felfogni. **A tesztelés során minden esetben a megrendelővel egyeztetett technikát kell alkalmazni, ám jelen tananyag nem tűzte ki azt a célt, hogy a vizsgálat garanciális feltételeit meghatározza.** ([11])

Az alábbiakban a SE felmérésre vonatkozóan az alapvető módszereket tekintjük át, azzal, hogy a módszerek segítenek azonosítani azt a pontot, amelyek kihasználásával hamarabb célt érhetünk. ([10])

## 3.1 Hálózati forgalom vizsgálata

A felhasználók internetezési szokásainak vizsgálatára az egyik legegyszerűbb eszköz a hálózati forgalom vizsgálata, azaz annak elemzése, hogy milyen weboldalakat látogatnak meg, milyen online szolgáltatásokat vesznek igénybe a vizsgált személyek.

A gyűjtött információkból következtetés vonható le arra vonatkozóan, hogy a felhasználó milyen mértékben van kitéve a külső támadásoknak, például kártékony programok jelentette veszélyeknek.

Az ily módon történő adatgyűjtéshez mindenképpen szükséges a felhasználók beleegyezése, hozzájárulása és a felmérésben résztvevők megfelelő tájékoztatása. A módszer hátránya, hogy amennyiben a felhasználó értesül arról, hogy internetes tevékenysége megfigyelésre kerül, előfordulhat, hogy változtat szokásain (például ezentúl nem látogat meg a munkahelyén szexuális tartalmú oldalakat), ami nem pontos eredményekhez vezet, és ezáltal torzítja a valós képet.

## 3.2 Social engineering audit

A legpontosabb felmérési eredmények eszköze egy social engineering audit lefolytatása, amely kifejezetten a kártékony programok terjesztési módszereinek vizsgálatára, illetve a felhasználók különféle, az emberi tényezőt kihasználó megkereséseivel szembeni ellenállóképességére helyezi a hangsúlyt.

Természetesen, mint minden más audit, a social engineering jellegű vizsgálatok is egy projekt keretein belül zajlanak. Mivel az ilyen jellegű sebezhetőségvizsgálat – tekintve, hogy a vállalat alkalmazottai „ellen" irányul, vagy legalábbis sokan tévesen így fogják fel – különösen „kényes" kategóriába tartozik, nagyon fontos a projekt megfelelő előkészítése mind a megbízó, mind a kivitelező szempontjából.

Megbízó oldalról még a vizsgálatok megkezdése előtt nagyon fontos tisztázni, hogy mely területek, személyek tartozhatnak a vizsgálat hatókörébe. Jellemző probléma annak dilemmája, hogy a vezetőség alávethető legyen-e a vizsgálatoknak. A vezető pozícióban elhelyezkedő személyek is különféle, akár specializált social engineering jellegű támadások célpontjaivá válhatnak (gondoljunk a korábban említett whaling támadásra), ezért mindenképpen javasolt, hogy valamilyen formában a vezetőség biztonságtudatossága is kerüljön vizsgálatra. Persze a vélemények megoszlanak, vannak, akik szerint nem célravezető, ha magasabb pozícióban levő személyek is áldozatául esnek valamilyen támadásnak, hiszen ha „elbuknak" a vizsgálaton, az nem biztos, hogy ösztönzi az alkalmazottakat a biztonságtudatos viselkedésre. Ebben ugyan van némi igazság, de nem ilyen szemszögből érdemes megközelíteni a dolgot, hiszen egy vezető pozícióban levő személy viselkedése gyakran példaértékű is lehet – tehát, ha hibázik, de megpróbál változtatni ezen, akkor valószínűleg az alkalmazottak is követni fogják a példáját.

Nemcsak konkrét személyeket, hanem egyes területeket (osztályokat, részlegeket) is kiemelten be lehet vonni, illetve ki lehet zárni a vizsgálatokból. Az informatikai részleget általában nem célszerű social engineering technikákkal megkeresni, hiszen – feltételezések szerint – ezen kollégák az átlag felhasználóktól jóval biztonságtudatosabbak. Amennyiben mégis az a döntés születik, hogy a rendszergazdák is kerüljenek tesztelésre, számolni kell annak veszélyével, hogy észlelik a gyanús csatolmányt, vagy a leadott talált adathordozóra írt fájlt azonosítják gyanús programként, és felhívják a többi munkatárs figyelmét a támadásra – így a teszt bizonyos értelemben "sikertelen" lesz.

A vizsgálandó területeken és személyeken túl meg kell határozni a vizsgálat hatókörét is, azaz pontosan meddig lehet elmenni egy-egy feladat végrehajtása során. Nem lenne ugyanis jó, ha egy támadás kivitelezésekor a vizsgálatot végző személy olyan mélységekre hatolna, mely már sértené a megbízó vállalatnak, vagy alkalmazottjának az érdekeit.

Ugyanakkor nem szabad megfeledkezni a vizsgálatot végző személyek védelméről sem, mert amennyiben tevékenységük lelepleződik, annak akár büntetőeljárás is lehet a végkifejlete. Ennek elkerülése érdekében biztosítani kell egy, a megbízó jogosult képviselője által aláírt, úgynevezett „Támogató nyilatkozatot", melyben a megbízó biztosítja, hogy a vizsgálatot végző személyek által elkövetett cselekményekről tudomásuk van, s ez a dokumentum mentesíti a biztonsági és hatósági eljárások alól.

# 3.3 Kérdőíves felmérés

A biztonságtudatosság tesztelésének egyik leghatásosabb módszere természetesen a korábban bemutatott social engineering audit lefolytatása. Ekkor ugyanis többnyire a néhány kiválasztott személy, illetve bizonyos támadásoknál a vállalat összes munkatársa „tesztelésre" kerül, s így meg lehet állapítani, hogy egy valós támadásnál mennyire állnák meg a helyüket a kiválasztott alkalmazottak.

A hangsúly azonban most a kiválasztott szón van, ugyanis mi van abban az esetben, ha egy olyan személyt sikerül kiválasztani, aki történetesen rendelkezik kellő figyelemmel vagy tapasztalattal, hogy hárítani tudja a támadást? Attól, hogy egy felhasználó biztonságtudatosan viselkedik, sajnos nem feltételezhetjük azt, hogy más felhasználó is ugyanilyen gondossággal végzi a munkáját, és egy hasonló átverés során ugyanilyen helyesen jár el.

A kártékony programok észlelésének tesztelésekor ugyan járható út, hogy a vállalat minden munkatársa mindenféle szimulált támadásnak (adathalászat, fertőzött csatolmány, beküldött adathordozó, stb.) célszemélyévé váljon, azonban a biztonságtudatosság szintjét, az információbiztonsági és egyéb belső szabályzatokban foglaltak ismeretét más módon is fel lehet mérni, mely hatékonyan kiegészítheti a social engineering audit során tapasztaltakat. A felmérés egyik hatásos módszere a kérdőíves megkeresés lehet, azaz töltessünk ki a szervezet minden munkavállalójával egy, a biztonságtudatosság szintjét felmérő kérdőívet.

Mit is kérdezzünk meg a felmérés során? Ezt természetesen minden vállalatra testreszabottan célszerű elkészíteni, általánosságban azonban elmondható, hogy olyan, a korábban már említett szabályzati pontok ismeretére kell rákérdezni, melyek megszegésével egy social engineer vissza tud élni, mely kihágásokat ki tudja használni. Annak érdekében, hogy a kitöltő személyek őszintén válaszolhassanak, célszerű a kérdőív anoním módon történő leadását biztosítani, így senkinek sem kell tartania a szankcióktól, ha esetleg bevallja, szándékosan a munkahelyi számítógépén tölt le illegális tartalmakat. Emellett, hogy ne csak vizsga érzetet keltő kérdéseket tegyünk fel, alkalmazhatunk néhány rövid esettanulmányt, melyhez kérdéseket kapcsolunk, illetve megkérdezhetjük a válaszadótól, hogy került-e már hasonló szituációba, találkozott-e valamilyen hasonló jellegű megkereséssel.

***Példa kérdés**:*

*Ön egy 32 GB-os pendrive-ot talál ottfelejtve a folyosón levő központi nyomtatón. Mit tesz a talált eszközzel?*

Semmit, majd észreveszi, aki otthagyta, vagy valaki más megtaláló majd megkeresi a gazdáját, én nem bajlódok ilyenekkel.

*a)* Leadom az érintett terület titkárságán, vagy írok egy kör e-mailt, hogy találtam egy adathordozót, de nem nézem meg, mi van rajta, mert nem érdekel.

*b)* Megnézem a tartalmát, hátha rájövök, kié lehet, és akkor visszaadom neki.

*c)* Megnézem a tartalmát, hátha van rajta valamilyen érdekes információ, lehet, hogy a pénzügyesek vesztették el… Vagy valami jó film van rajta…

*d)* Pont szükségem van egy pendrive-ra, ez a 32 GB-os darab épp jókor került elvesztésre.

*e)* Leadom a pendrive-ot az információbiztonságért felelős területnek, ők majd kiderítik, mi van a pendrive-on és kié lehet.

*f)* Megjegyzendő, hogy ahogyan a fenti példában is látszik, a kérdésekre adott válaszok, lehetséges szituációk közül nem feltétlenül kell egyetlennek helyesnek lennie, hiszen az embert több dolog is motiválhatja, például egy talált adathordozó tartalmának megtekintésére.

A kérdőívet úgy kell összeállítani, hogy olyan kérdések és esettanulmányok kerüljenek bele, melyek a vizsgált szervezetnél relevánsak, nem fog fennállni annak az esete sem, hogy valaki találomra válaszol, hogy vele ilyen úgysem fog előfordulni. Fontos ezért, hogy a kérdések és esettanulmányok összeállításakor mindenképpen szerezzünk elegendő információt a megbízótól, használjuk fel a belső szabályzatait is.

A kérdőív lehet hagyományos papír alapú, vagy a ma már sokkal népszerűbb online kitölthető verzió is. Online kérdőívnél megvalósítható azon probléma kiküszöbölése is, hogy a felhasználó ne tudja kiválasztani a legjobb megoldást a lehetséges válaszok közül, ugyanis beállítás szerint a kérdésre feltett válaszok egyesével, egymás után jelennek meg, és csak elutasítani vagy elfogadni tudja azt. Utóbbi azért is lehet jó megoldás, mert így esetleg lehetőség van egy olyan kérdőívkitöltő program megalkotására, mely az adott válasznak megfelelően azonnal visszajelzést, jó tanácsot

is ad a felhasználónak. Nagyon fontos tudatni a munkatársakkal, hogy a kérdőív őszintén történő kitöltése az ő érdekük, hiszen a felmérésnek csak akkor van értelme, ha mindenki az igazat válaszolja.

A beérkezett válaszok alapján ki lehet alakítani a vizsgált szervezet munkatársairól egy olyan képet, hogy mely területeken bizonyulnak saját bevallásuk szerint a „leggyengébbnek", esetleg mely szabályzati pontok azok, amelyeket a felhasználók nem értenek, vagy nem szeretnek betartani. Mindezek nagyon hasznos alapot nyújthatnak a szabályzatok elkészítésekor, módosításakor, valamint kiderül az is, a biztonságtudatossági oktatásokon mely területekre kell fektetni a hangsúlyt.

Javasolt, hogy a social engineering audit feladatok lefolytatása után a megbízó minden munkatársával kerüljön kitöltetésre egy ilyen, a szervezetre szabottan készült, kártékony programok észlelésével kapcsolatos biztonságtudatossági kérdőív. Ennek eredményeit esetleg összevetve a tesztek során történtekkel érdekes eredményeket is kaphatunk, valamint a biztonságtudatossági oktatások tananyagának elkészítésében is hasznos segítségnek bizonyulhatnak.

# 4. IRODALOMJEGYZÉK


[1] Esettanulmány egy felfedezett poloskaprogramról, SaveAs Kft., 2002.

[2] Mitnick, K. D. és Simon, W. L. (2003): A legendás hacker – A megtévesztés művészete, Perfact Kiadó, Budapest

[3] Trend Micro: The Custom Defense Against Targeted Attacks, A Trend Micro White Paper http://www.trendmicro.com/cloud-content/us/pdfs/business/white-papers/wp_custom-defense-against-targeted-attacks.pdf

[4] Harl, G. (1997): People Hacking – The Psychology of Social Engineering, http://www.psihoworld.co.ba/The%20Psychology%20of%20Social%20Engineering.pdf

[5] von Neumann, John: *The Theory of Self-reproducing Automata*, A. Burks, ed., Univ. of Illinois Press, Urbana, IL, 1966

[6] Nmap Network Scanning, Chapter 8. Remote OS Detection, www.insecure.org/nmap/nmap-fingerprinting-article.html

[7] RFC 793, TRANSMISSION CONTROL PROTOCOL, http://www.ietf.org/rfc/rfc793.txt

[8] Közösségi webhelyek információbiztonsági kockázata, http://gazdasagiradio.hu/cikk/8566/

[9] http://www.googleguide.com/advanced_operators.html

[10] Oroszi, E.: Social Engineering – Az emberi erőforrás, mint az információbiztonság kritikus tényezője, diplomadolgozat, Budapesti Corvinus Egyetem, 2008

[11] Krasznay, Cs.: A magyar elektronikus közigazgatási alkalmazások információbiztonsági megoldásai, doktori értekezés, ZMNE, Hadtudományi Kar, Katonai Műszaki Doktori Iskola

[12] Herzog, P.: Open-Source Security Testing Methodology Manual 2.2., Institute for Security and Open Methodologies.

[13] Scarfone, K.: NIST Special Publication 800-115: Technical Guide to Information Security Testing and Assessment, Gaithersburg, National Institute of Standards and Technology.

[14] Meucci, M. OWASP Testing Guide V3.0. 2008.: Open Web Application Security Project.

[15] Common Criteria Development Board: Common Criteria for Information Technology Security Evaluation: Evaluation methodology, Version 3.1, revision 3.

[16] The Open Web Application Security Project: OWASP Application Security Verification Standard 2009. (pp.: 4-16.).

[17] McClure, S., Scambray, J., Kurtz, G.: Hacking Exposed: Network Security Secrets and Solutions, Sixth Edition. McGraw-Hill Osborne Media. ISBN: 0071613749

[18] Mitnick, K.: 2-day Social Engineering Training Course Outline, Mitnick Security Consulting, LLC, http://mitnicksecurity.com/media/msc_course_outline.pdf

[19] Guenther, M.: Social Engineering Security Awareness Series.

[20] The Open Web Applications Security Project: OWASP Code Review Guide.

[21] https://www.us-cert.gov/government-users/compliance-and-reporting/incident-definition

Nemzeti
Közszolgálati Egyetem
Vezető- és Továbbképzési Intézet

SZŐKE GERGELY LÁSZLÓ

# A személyes adatok védelmének szabályozási környezete és gyakorlati kérdései

Budapest, 2014

A tananyag az ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel című projekt keretében készült el.


**Szerző:**
© Szőke Gergely László 2014

**Kiadja:**
© NKE, 2014

**Felelős kiadó:**
Patyi András
rektor

# Tartalom

# I. Az adatvédelmi szabályozás kialakulása[1]

## 1. Az adatvédelmi szabályozás generációi

Az adatvédelmi jog viszonylag rövid története ellenére a szakirodalom a szabályozás több korszakát, generációját különbözteti meg, amelyek különböző – de mindannyiszor a technológia fejlődéséhez, és az ehhez szorosan kapcsolódó társadalmi változásokhoz köthető – kihívásokra reagálva hasonló szabályozási célokat eltérő megközelítéssel igyekeztek megvalósítani.

Az adatvédelemmel foglalkozó szerzők álláspontja ugyanakkor nem egységes atekintetben, hogy pontosan hány generációt érdemes megkülönböztetni, és hogy az egyes nemzetközi, európai és nemzeti jogforrások pontosan melyik generációhoz tartoznak.

A magyar jogirodalomban először Majtényi László foglalja össze az adatvédelem korszakait, és három szabályozási generációt különböztet meg: az első generációs szabályok a 70-es években fejlődtek ki, és az állami, számítógépes (legalább részben automatizált) nyilvántartásokkal szemben igyekezett valamilyen védelmet kialakítani. A második generációs szabályok a 80-as, 90-es években jelentek meg, és már nem csak az automatizált, de a papíralapú nyilvántartásokat is a szabályozás hatálya alá vonták. Végül Majtényi szerint a harmadik generációs szabályok főbb jellemzői az európai integráció sajátosságainak figyelembe vétele, és a szektorális szabályok megjelenése.[2]

A magyar jogirodalomban az eddigi legrészletesebb elemzés az adatvédelem történetéről Jóri András munkáiban található. Jóri szintén három korszakot különböztet meg, de a 80-as, 90-es évek fejleményeit egy szabályozási generációhoz sorolja, és a harmadik generációs szabályozás egyes elemeinek megjelenését a német Teledienstedatenschutzgesetz (TDDSG) 1997-es megalkotásához köti.[3]

Végül Majtényi felosztásához nyúl vissza később Hegedűs Bulcsú is, aki azonban egyrészt a második generációs szabályozás fő jellemzőjeként azt emeli ki, hogy annak hatálya már az üzleti élet gyakran igen adatéhes szereplőire is kiterjed, másrészt nála is megjelenik egy újabb – negyedik – generációs szabályozás gondolata, amelynek főbb jellemzői az önszabályozás, az Internettel kapcsolatban megjelenő adatvédelmi kérdések és a magánszférát erősítő technológiák megjelenése.[4]

E rövid áttekintésből az látható, hogy több közös pont ellenére sem lehetséges egyértelműen a generációk közötti korszakváltások határát kijelölni, vagy az egyes jogforrásokat egyértelműen egyik vagy másik generációhoz tartozónak tekinteni. A generációs felosztásnál valójában nagyobb jelentősége van az egyes szabályozások főbb jellemzőit és a fejlődési tendenciákat kutatni.[5] Ugyanakkor mivel a megfelelő korszakolás mégiscsak jól átláthatóvá teszi az elemzést, indokoltnak tartjuk az adatvédelem történetét e logikai séma mentén bemutatni.

A generációk vagy korszakok számának meghatározásakor a magunk részéről azt gondoljuk, hogy csak a valóban jelentős, koncepcionális kérdésekben is újítást hozó változások esetén érdemes új generációs szabályozásról beszélni. Alapvetően így Jóri András megközelítését követjük, amely szerint az eddigiek során alapvetően két különböző szabályozási generáció különböztethető meg egyértelműen azzal, hogy az elmúlt évtizedben megjelentek olyan új tendenciák, amelyek egyértelműen egy új, harmadik generációs adatvédelmi szabályozás kialakulása felé mutatnak.

A kutatás egyik tétele éppen az, hogy az elmúlt egy-másfél évtizedben bekövetkezett technológiai és társadalmi változások következtében az adatvédelmi szabályozás újra megérett az átfogó reformra, és olyan új szabályokra van szükség, amelyek az eddigiektől jelentősen eltérő hangsúlyokat állapítanak meg az adatvédelem területén. Az Európai Bizottság adatvédelmi rendelettervezete nagyjából megfelel e kritériumoknak, így e dokumentumot, (pontosabban annak a kézirat lezárásakor fellelhető szövegverzióját) a harmadik generációs szabályozás „mintaszabályozásának" tekintjük, és ennek fényében elemezzük.

Az alábbiakban áttekintjük az adatvédelmi szabályozás három korszakát. Az egyes szabályozási generációk áttekintését az adott kor technológiai és társadalmi hátterének rövid bemutatásával kezdjük, az adatvédelmi szabályozás megértéséhez ugyanis szükségesnek látszik ennek vizsgálata is. A technológia jellege, költségei és elérhetősége nagymértékben meghatározzák azok felhasználóit, és így az adatkezelések alanyait is – míg e két tényező együttesen befolyásolja az adatkezelésekkel járó, magánszférára gyakorolt potenciális veszélyeket. A társadalmi közgondolkodás – Európában elsősorban az információs társadalommal kapcsolatos diskurzus – pedig azt határozza meg, hogy

1 Jelen fejezet Szőke Gergely László: Az adatvédelem szabályozásának történeti áttekintése (Infokommunikáció és jog, 2013/3) című tanulmány alapján készült.
2 Majtényi László: Az információs jogok. in.: Halmai Gábor – Tóth Gábor Attila (szerk.): Emberi jogok. Osiris, 2003, 582-583.
3 Jóri András: Adatvédelmi kézikönyv. Osiris, 2005, 24-66.
4 Hegedűs Bulcsú: Az adatvédelmi jog általános tanai. In.: Tóth András (szerk.): Infokommunikációs jog II. Patrocínium, 2013, 137-145.
5 Egy-egy, adott szabályozási modellnél kifejtett jellemző rendszerint jóval korábban megjelenik a jogirodalomban.

e jelenségekre a társadalom, előbb-utóbb a jogalkotás eszközével (is) élve, illetve a jogszabályok végrehajtása során, miként reagál.[6] Ezt követi az adott korszak adatvédelmi szabályozásának, jellemzőinek bemutatása és rövid elemzése.

Mielőtt azonban rátérnénk az európai adatvédelmi szabályozás 1970-től tartó történetére, érdemes vázlatosan felidézni a magánszféra-védelem néhány korábbi sarokpontját is.

## 1.1 Előzmények

Amint azt említettük, az adatvédelmi jog fejlődését az 1970-es évekre kibontakozó technológiai forradalomra adott válaszlépésként értékeljük. Érdekesség, hogy más az első magánszféra-védelemmel foglalkozó tanulmány, Samuel D. Warren és Louis D. Brandeis sokat hivatkozott,[7] 1890-ben a Harvard Law Review hasábjain megjelent, „The Right to Privacy" című tanulmánya[8] is alapvetően az adott kor technológiai és társadalmi változásaira reagált. A Kodak 1888-ban piacra dobta a fényképezés történetében mérföldkőnek számító Kodak 1 fényképezőgépét, amely egyrészt széles körnek tette elérhetővé a fényképezőgép használatát, másrészt lehetővé tette az azonnali – az alany hosszas egyhelyben maradását nem igénylő – fényképkészítést.[9] A szerzők emellett kiemelik a sajtó, különösen a bulvársajtó fokozódó szerepét, és arra jutnak, hogy e jelenségekre válaszul szükséges lenne egy új jog, a magánszférához való jog elismerése.[10]

Később elsősorban az állami információs túlhatalommal szembeni félelmek jelentek meg, eleinte a szépirodalomban, később a társadalomtudományi és jogi szakirodalomban is. Előbbi területen kétségtelenül George Orwell 1984 című regénye a legismertebb, amelyben a totalitárius állam a „telekép" technológiáját használja az emberek folyamatos megfigyelésére. Az orwelli „Nagy Testvér" kifejezés aztán széles körben vált a megfigyelő és elnyomó állam jelképévé. E fenyegetéshez aztán később csatlakoztak az üzleti élet „Kis Testvérnek" nevezett szereplői is.

Az információs jogok teljes hiányának állapotát igen nyomasztóan érezheti át az olvasó Franz Kafka: A per című művének végsőkig kiszolgáltatott főhősének (Josef K.) történetén keresztül is. A magyar költészetben ugyancsak megjelenik a „levegőtlenség érzése": József Attila Levegőt! című versének sorai világos és hatásos kifejezőereje miatt gyakran idézettek a magyar adatvédelmi szakirodalomban is:[11]

„Számon tarthatják, mit telefonoztam/ s mikor, miért, kinek./ Aktába írják, miről álmodoztam,/ s azt is, ki érti meg./ És nem sejthetem, mikor lesz elég ok,/ előkotorni azt a kartotékot,/ mely jogom sérti meg."[12]

A II. világháború tapasztalatai is különös óvatosságra intettek. A háború borzalmai és a náci rémtettek nemcsak az emberi jogok elfogadásának és nemzetközivé válásának adott új lendületet,[13] de a (hatékony) állami nyilvántartásokkal és a modern technológiával való visszaélések lehetőségére is rámutattak.

A lyukkártya-technológia az 1935-ös és 1939-es német népszámlálások kiszolgálása mellett a háború logisztikáját, a zsidóüldözés különböző formáit és a holokauszt szervezését is hatékonnyá tették.[14] Az IBM és – a később állami kézbe vett – német leányvállalata, a Dehomag[15] által kínált technológia (és a hozzá tartozó komoly és folyamatos technikai támogatás) hozzájárult a zsidó népesség faji alapú megszámlálásához és azonosításához, javaik felméréséhez, és a zsidó kötődésű vállalkozások számbavételéhez is (amely e javak elkobzásához és a zsidónak tekintett vállalkozások államosításához vezettek).[16] Ugyanez a technika szolgálta ki holokauszt megszervezését is, a vasúti menetrendek op-

6 A technológia történetét és a társadalmi változásokat nem a teljesség igényével elemezzük, hanem csak azokat a jelentségeket mutatjuk be, amelyek az adatvédelmi szabályozásra alapvető hatással voltak.

7 A tanulmány egyes elemeit és hatását elemzi például Jóri András, im. 14-15, Majtényi László: Az információs szabadságok. Complex, 2006, 28-30., Sólyom László: A személyiségi jogok elmélete. Közgazdasági és Jogi Könykiadó, 1983 201-211. Az Információs Társadalom című folyóirat 2005/2. számában teljes tanulmányt szán a témának Simon Éva (Simon Éva: Egy XIX. századi tanulmány margójára, Információs Társadalom 2205/2, 32-43). Ugyanitt megjelent az eredeti tanulmány magyar nyelvű fordítása is.

8 Warren, Samuel D. – Brandeis, Louis D: The Right to Privacy, Harvard Law Review 1890/4. http://heinonline.org/HOL/Page?collection=journals&handle=hein.journals/hlr4&id=205&terms=photograph#207 [2013.06.15.] 195-220.

9 A Kodak szlogenje szerint „ön megnyomja a gombot, a többi a mi dolgunk". A cég ugyanis vállalta a fényképek előhívását is, amelyet korábban csak megfelelő szakértelemmel rendelkező fényképészek tudtak megtenni. History of Kodak, http://www.kodak.com/ek/US/en/Our_Company/History_of_Kodak/Milestones_-_chronology/1878-1929.htm [2013.08.22.]

10 Warren, Samuel D. – Brandeis, Louis D im. 195.

11 A szépirodalom és az adatvédelem kapcsolatát ld. pl. Balogh Zsolt György: Jogi informatika, Dialóg Campus, Budapest-Pécs, 1998, 152-153., Jóri András im. 21-22. A kérdéssel legrészletesebben Majtényi László foglalkozik (Majtényi László: Az információs szabadságok, 41-55.), aki nemcsak, hogy maga is keresi a „Nagy Privacy Metaforát" (és találja meg végül József Attila idézett versében), de a nemzetközi szakirodalomban is népszerű metaforakeresés okait is kutatja: „Olyasmiről beszélünk ugyanis, amiről nem tudjuk, hogy micsoda. Mindenki, aki a privacyvédelemmel foglalkozik, valamelyest szenved attól, hogy a védelem tárgya, alanya bár megnevezhető, meghatározhatatlan. Ezért menekülnek a képes beszédhez. A szerzők mindegyike valami sipolyt keres a jogi burkon, hogy azt megnyitva, meglesse az ént." Majtényi László: Az információs szabadságok, 46.

12 József Attila: Levegőt József Attila összes versei. Szépirodalmi Könyvkiadó, Budapest. 1966. 445. o.

13 Kardos Gábor: Az emberi jogok nemzetközivé válása. In.: Halmai Gábor – Tóth Gábor Attila (szerk.): Emberi jogok, Osiris, Budapest 2003, 67.

14 Galántai Zoltán: E-privacy olvasókönyv, 2003. http://mek.oszk.hu/04100/04134/html/ [2013.07.17.]

15 Deutsche Hollerith-Maschinen Gesellschaft mbH

16 Black, Edwin: Az IBM és a Holokauszt, Atheneum 2000, Budapest 2002, 90-91.

timalizálásától a koncentrációs táborok adminisztrációjáig bezárólag.[17] Emellett a náci Németország nagymértékben támaszkodott a lyukkártyarendszerre a háború és az utánpótlás szervezésében is.[18]

A hollandiai zsidóság nagyarányú elhurcolását ugyancsak a hatékony és átfogó népesség-nyilvántartás tette lehetővé, amelyet – lyukkártya-technológiával támogatva – az 1930-as években hoztak létre az állami feladatok hatékonyabb ellátása érdekében. A nyilvántartás megalkotásakor a holland állampolgárok még joggal bíztak a kormányzatukban – a náci megszállással járó veszélyekkel azonban nem számoltak. Míg Hollandiából a zsidóság 73%-át vitték el a németek, addig Franciaországból csak 25%-öt.[19] Franciaországban ugyanis a korábbi népszámlálások során nem kérdeztek rá a vallási hovatartozásra, így kész nyilvántartás nem állt rendelkezésre.[20] Emellett a lyukkártyarendszert a közigazgatásban csak jóval kisebb mértékben sikerült elterjeszteni, mint Hollandiában vagy Németországban, így a gyors népesség-összeírási kísérletek is döntően kudarcba fulladtak.[21]

Magyarországon az 1941-es népszámlálás adatait használták fel a német nemzetiségű állampolgárok világháborút követő kitelepítése során:[22] „Németországba áttelepülni köteles az a magyar állampolgár, aki a legutolsó népszámlálási összeírás alkalmával német nemzetiségűnek vagy anyanyelvűnek vallotta magát".[23] A KSH végül az elfogadott jogszabályok alapján kénytelen volt együttműködni a kitelepítést lebonyolító szervekkel.[24] Megjegyzendő, hogy ez volt az első olyan népszámlálás, amely nem az anyanyelvre, hanem a nemzetiségre kérdezett rá – heves vitát kiváltva a statisztikusok körében arról, hogy ilyen szubjektív körülményt a statisztika tudománya egyáltalán használhat-e.[25]

Ezek az információk, legalábbis olyan alaposan feldolgozva, mint ahogyan jelenleg hozzáférhetőek, a 60-70-es években – az adatvédelmi diskurzus kezdetén – ugyan nem feltétlenül minden esetben álltak rendelkezésre, de a történelmi tapasztalatokra való utalás gyakran megjelenik a szakirodalomban.[26]

## 1.2 Az első generációs adatvédelmi szabályozás kialakulása és jellemzői

Az adatvédelmi szabályozás első generációjának megjelenése az 1970-es évekre tehető, és alapvetően – csakúgy, mint eredetileg Warren és Brandeis cikke, és csakúgy, mint azóta több, adatvédelmet érintő jelentős változás – a technológia fejlődéséből eredő veszélyekre adott jogi válaszlépésnek tekinthető.

### 1.2.1 Technológiai és társadalmi háttér

A számítástechnika fejlődése az 1960-as évek végére eljutott arra a szintre, hogy reális lehetőséggé váljon az állami nyilvántartások adatainak elektronikus tárolása, és a nyilvántartásokban való gyors keresés. Az adatokkal való visszaélések – a papír alapú elkülönített adatbázisok fizikai jellemzőinél fogva fennálló – természetes korlátai ledőltek. A papír alapú adatbázisok ugyanis fizikai határt szabtak az adatkezelőnek a tekintetben, hogy mennyi adatot képes kezelni. A papír tömege, az átláthatóságot biztosító nyilvántartási rendszer költséges volt, és a sokszor különálló adatállományokban való keresés időigényes, megfelelő katalogizáltság nélkül pedig szinte lehetetlen volt.[27]

Az automatizált adatfeldolgozásra való igénnyel párhuzamosan Európa szerte megjelentek a különböző állami nyilvántartások egységesítésének vagy legalábbis összekapcsolásának tervei is.[28] Később, az információs társadalomról szóló diskurzus során a centralizáció-decentralizáció kérdése egyébként igen hangsúlyos szerepet kap – mindkét irány

17 Black, Edwin, im. 23-25. 200-201. 256.
18 Black, Edwin im. 157-159.
19 Mayer-Schönberger, Viktor: Delete: The Virtue of Forgetting in the Digital Age, Princeton University Press, Princeton and Oxford 2009, 141.
20 Black, Edwin im. 236.
21 „Sem a franciák, sem a németek nem tudták pontosan kideríteni, hogy az elkövetkezendő hónapok, sőt a hátralévő háborús évek alatt ki, milyen módszerrel végzett népszámlálást az országban." Black, Edwin im. 238. A holland és francia rendszer összehasonlítását ld. részletesen Black, Edwin im. 222-250.
22 Hegedűs Bulcsú im. 130.
23 A 12330/1945. ME rendelet 1. §-át idézi Dobos Balázs: A magyarországi németek kitelepítése az 1941-es népszámlálás tükrében, http://www.hhrf.org/kisebbsegkutatas/kk_2005_03/cikk.php?id=954 [2013.07.17.]
24 A példa jól megvilágítja azt is, hogy milyen jelentős különbség van a között, hogy egy szervezet nem jogosult (az éppen hatályos jogszabályok szerint) bizonyos adatok átadására, és a között, hogy a szerv nem is rendelkezik az adott személyes adatokkal (mert például az adatokat anonimizálták), és így nem is képes személyes adatok átadására. Az ezzel kapcsolatos dilemmák, pro és kontra érvek a mai napig megjelennek az adatvédelemről szóló diskurzusban. Ezt a problémakört Majtényi a ruandai polgárháború példájával illusztrálja, ahol az uszítók rádióban olvasták be a meggyilkolandók neveit – „mégsem mindegy az sem, hogy a még valamennyire konszolidált hatalom (mely mindig csak viszonylagosan normális) milyen adatbázisokat hagy tébolyult utódjára." Majtényi László: Az információs szabadságok, 80. A holland zsidók tragédiája ugyanezt támasztja alá.
25 Dr. Heinz Ervin – Dr. Lakatos Miklós: A Központi Statisztikai Hivatal szerepe a német lakosság kitelepítésében. in.: Czibulka Zoltán – Dr. Heinz Miklós – Dr. Lakatos Miklós (összeáll.): A magyarországi németek kitelepítése és az 1941. évi népszámlálás. Központi Statisztikai Hivatal Népszámlálási Főosztálya, Budapest, 2004 2-3.
26 Ld. pl. Burkert, Herbert: Privacy - Data Protection. A German/European Perspective, 1999 http://www.coll.mpg.de/sites/www.coll.mpg.de/files/text/burkert.pdf [2013.07.17.] 49-50.
27 Hegedűs Bulcsú im. 133.
28 Burkert, Herbert im. 44-51.

mellett komoly érvek hozhatók.[29] A költségcsökkentés és az államok „természetes" központosító törekvései azonban ebben az időszakban alapvetően a centralizált rendszerek felé mutattak.

A nyilvántartások egységesítése és/vagy egy nagy, mindenre kiterjedő központi (népesség)nyilvántartás létrehozása és működtetése a legegyszerűbben valamilyen egységes személyi azonosító használatával lehetséges,[30] így több államban is kísérletet tettek ezek bevezetésére. E törekvések alapvetően az egyre több feladatot magára vállaló jóléti állam információigényét hivatottak kiszolgálni.

A technológia magas költségei miatt annak használatát csak a nagy adatkezelők, elsősorban az állam különböző szervei és néhány nagyvállalat engedhette meg magának. A technológia fejlettsége tehát közvetlenül meghatározta az azt felhasználók, és így a potenciális adatkezelők körét is: ez államonként néhány, elsősorban állami szervet jelentett, így a szabályozás is erre a körre koncentrált.[31]

A számítástechnika fejlődésének társadalomra gyakorolt hatása a társadalomelmélet képviselőit is foglalkoztatni kezdte, és nagyon korán megjelentek az elektronikus adatfeldolgozás magánszférára gyakorolt hatásairól szóló felvetések is.[32] Az 1970-es években napvilágot láttak az információs társadalom kialakulásáról szóló diskurzus első csírái is[33] – hogy aztán ez a 80-as években kiteljesedjen. 1973-ban jelent meg Daniel Bell iskolateremtő műve a posztindusztriális társadalomról.[34] A műben Bell – az információs társadalom kifejezést a posztindusztriális társadalom helyettesítőjeként használva – egy olyan szolgáltató társadalom eljövetelét vizionálja, amelyben a rendszerezett elméleti tudás és az ezzel együtt járó innovációs készség jelentik a társadalom meghatározó stratégiai erőforrását.[35]

### 1.2.2 Az első adatvédelmi törvények elfogadása

A fenti technikai-társadalmi háttér ismeretében talán nem meglepő, hogy élénk vita alakult ki, amikor a németországi Hessen tartomány egységes népesség-nyilvántartó adatbázis kialakítását kezdte meg. A „Nagy Hesseni Terv" című előkészítő dokumentumot áthatja – Sólyom találó kifejezésével élve – a „technokrata aggálytalanság". A kormányzat a társadalom államosítását és funkcionálisan integrált igazgatási rendszert vizionál – ahol is a „statisztikai hivatal a rendőrséggel, iskolával, orvossal" kommunikál. Az egységes, az egyént személyi számmal azonosító adatbázisban mintegy 70 információt terveztek tárolni.[36] A terv végül egészen más formában valósult meg, és elfogadásra került Európa első adatvédelmi törvénye, amely döntően meghatározta az elkövetkezendő adatvédelmi viták irányát Németországban és azon kívül is.[37]

A hessenihez hasonló tervek születtek Európa számos országában, több esetben hasonló vitát és megoldásokat generálva. Franciaországban szintén egységes azonosítószám alapján tervezték összevonni a meglévő nyilvántartásokat – ráadásul, igen szerencsétlen módon, titkos terv keretében, így itt egy 1974-es sajtócikk kényszerítette ki a társadalmi vitát. Egy másik, a gyermekeket születésüktől fogva nyilvántartó adatbázis a „problémás" gyermekek kiszűrését szolgálta volna. Találóan jegyzi meg Burkert, hogy „egy ilyen adatbank szimbolikus jelentősége – a nem túl távoli történelmi múlt fényében – drámai volt".[38]

Emellett 1973-ban Svédországban, 1977-ben a Német Szövetségi Köztársaságban (immár szövetségi szinten), 1978-ban Dániában, Norvégiában, Ausztriában és az imént bemutatott folyamatok lezárásaként Franciaországban fogadtak el adatvédelmi törvényeket.[39]

29 A „nagy rendszerek" hívei azok optimális kihasználtságát és így a fajlagos költségek csökkentését, valamint a szabványosításban rejlő előnyöket hangsúlyozták. A decentralizált, de együttműködő rendszereket támogató szakértők a rossz értelemben vett uniformizálást és a szabadságjogok fenyegetését hozták fel ellenérvként, vitatva egyébként a költségcsökkentő hatást is (mivel a centralizált rendszerekhez lényegesen drágább hardver és szoftver, az üzemeltetéshez pedig felkészült szakembergárda szükséges). Balogh Zsolt György im. 154-155.

30 Ld. Jóri András im. 24., illetve ahogy Sólyom fogalmaz: „a rendszernek velejárója az emberek megszámozása", Sólyom László: Egy új szabadságjog: az információszabadság. Valóság, 1988/9. 25.

31 Jóri András im. 24.

32 Ld. például Arthur Millernek a Michigan Law Review folyóiratban 1969-ben megjelent írását: Miller, Arthur: Personal Privacy in the Computer Age: The Challenge af a New Technology in an Information-Oriented Society. In.: Michigan Law Review 1968-1969/67, 1089-1246. http://heinonline.org/HOL/Page?handle=hein.journals/mlr67&div=76&collection=journals&set_as_cursor=2&men_tab=srchresults#1103 [2013.06.15.] 1969, különösen 1107-1109.

33 Ld. például Brian Murphy 60-as években végzett kutatásainak összefoglalását Balogh Zsolt György im.. 150. Említhetjük még James Martin és Adrian R. D. Norman „The computerized society. An appraisal of the impact of commputers on society over the next fifteen years." című, 1970-ben megjelent munkáját.

34 Daniel Bell: The coming of postindustrial society: a venture in social forecasting, Basic Books, New York, 1973.

35 Balogh Zsolt György im. 150. valamint Hassan, Robert: The Information Society, Polity Press, Cambridge 2008, 52-53.

36 Sólyom László: Egy új szabadságjog, 25.

37 Simitis, Spiros: The Hessian Data Protection Act. The Hessian Data Protection Commissioner, Wiesbaden 1987, 5.

38 Burkert, Herbert im. 49-50., a szerző saját fordítása

39 Hegedűs Bulcsú im. 137.

### 1.2.3 Az első generációs szabályozás főbb jellemzői

Az első generációs szabályozást áthatotta a „Nagy Testvér" információs túlhatalmától való félelem, így e jogszabályok elsődleges célja a nagy (döntően állami) adatbázisok átláthatóságának megteremtése volt.[40] Ugyanakkor kezdetektől fogva felmerült, hogy a szabályozás hatálya kiterjedjen-e a nem állami adatkezelőkre. A hesseni törvény, inkább a tartományi hatáskörből, mintsem szigorú elvi megfontolásokból fakadóan, még csak az állami szervekre vonatkozó szabályokat tartalmazott, a német szövetségi adatvédelmi törvény azonban – épp e kérdés körül kialakult igen hosszas vita után – az állami- és magán adatkezelőkre egyaránt kiterjedt, csakúgy, mint az 1978-as francia és dán szabályozás.[41] Az első generációs szabályok alapvetően az automatizált adatkezelésekre terjedtek ki, és hangsúlyos szerepet kapott a konkrét technológia szabályozása.[42]

Ugyancsak fontos jellemző, hogy e törvények még nem biztosítottak általános rendelkezési jogot az adatalanyok számára a személyes adataik felett, de biztosítottak néhány részjogosítványt, például a betekintés és a helyesbítés jogát.[43]

Már e korai jogszabályok felismerték azt, hogy az adatvédelem érvényesülése csak megfelelő felügyelőhatóságok felállítása mellett biztosítható. A jogszabályok rendezték az ombudsman jellegű vagy hatósági hatáskörökkel (is) rendelkező felügyelőszervek feladat- és hatásköreit. Így a hesseni törvény létrehozta a Hesseni Adatvédelmi Biztos[44] intézményét, a francia jogszabály a kezdetektől fogva részletesen szabályozta a francia adatvédelmi hatóság (CNIL)[45] jogállását, míg Svédországban a Swedish Data Inspection Board végezte és végzi az adatvédelem felügyeletét.[46]

Az 1973-as svéd törvény vezette be azt a később általánossá váló kötelezettséget, miszerint az adatkezelők kötelesek egy nyilvánosan hozzáférhető hatósági nyilvántartásba bejelenteni az egyes adatkezeléseiket (adatbázisaikat). Ez egyrészt biztosította az állampolgárok és fogyasztók számára az adatkezelések átláthatóságát, másrészt segítette az adatvédelmi hatóságok jogalkalmazását.[47] Jóri ugyanakkor felhívja a figyelmet arra, hogy ez a jogintézmény egy olyan korban született, amikor ez a kötelezettség csupán néhány, de jelentős mértékű adatbázist kezelő adatkezelőre vonatkozó kötelezettség volt.[48]

## 1.3 A második generációs adatvédelmi szabályozás kialakulása és jellemzői

### 1.3.1 Technológiai és társadalmi háttér

Az 1980-as években megjelent és hamarosan elterjedt a személyi számítógép,[49] amely drasztikus változásokat hozott. A nagyteljesítményű számítógépes kapacitás a korábbinál lényegesen szélesebb kör számára vált elérhetővé: a számítógépek alkalmazása gyorsan elterjedt az üzleti életben, és a PC megjelent az otthonokban is.[50]

Az 1990-es években újabb jelentős fejlemény történt. Az Internet kereskedelmi célú megjelenése és elterjedése a korábbi, önálló egységként funkcionáló számítógépeket világméretű hálózattá kötötte össze. Minden korábbinál könnyebbé és gyorsabbá vált az adatok (ideértve a személyes adatokat is) továbbítása akár a világ távoli pontjára is.

Az informatikai és kommunikációs technológiák fejlődése alaposan átalakította az üzleti szférát is. A fogyasztók számára a leglátványosabb terület kétségtelenül a marketingeszközök változása: az új kommunikációs csatornán (elsősorban e-mailen) keresztül megvalósuló, és egyre kifinomultabb direktmarketing technikák, valamint a jellemzően cookie-k használatán alapuló online marketing megjelenése.[51] A háttérben azonban más változások is történtek: az ügyféladatok illetve a vállalkozás belső működését jellemző adatok elektronikus adatbázisba szervezése lehetővé tette olyan új ügyfélkapcsolati (CRM) és vállalatirányítási (ERP) rendszerek kialakítását, amelyek korábban elképzelhetetlenek voltak.

E tendenciák világossá tették, hogy az üzleti szektor, (a „Kis Testvér") adatéhsége legalábbis vetekszik az államéval. Ezáltal a korábbi néhány, „jól látható" adatbázist (és azok kezelőit) adatkezelők milliói váltották fel. Ezek egy része – például pénzintézetek, hírközlési szolgáltatók, stb. – ráadásul egy-egy érintettről is igen nagyszámú, az érintett magánszférája szempontjából érzékeny adatot kezeltek.[52] Olyan új szabályozásra volt tehát szükség, amely képes a kialakult helyzetet kezelni.

40 Jóri András im. 24.
41 Burkert, Herbert im. 47-50.
42 Jóri András im. 25.
43 Ezek később az információs önrendelkezési jog részjogosítványai lettek, ld. Jóri András im. 24.
44 Der Hessische Datenschutzbeauftragte
45 Commission nationale de l'informatique et des libertés
46 Burkert, Herbert im. 46. 50-51. Csupán néhány példát emeltünk ki, a további nemzeti hatóságok felsorolásától eltekintünk.
47 Burkert, Herbert im. 48.
48 Jóri András im. 25.
49 Personal Computer, bevett rövidítéssel: PC
50 Hassan, Robert im. 3.
51 Turow, Joseph – Draper, Nora: Advertising's new surveillance ecosystem. in.: Ball, Kirstie – Haggerty, Kevin D. – Lyon, David (eds.): Routledge Handbook of Surveillance Studies. Routledge, London, 2012 134-135
52 A "Kis Testvér" előretöréséről ld. pl. Majtényi László: Az információs szabadságok, 36., Hegedűs Bulcsú im. 137.

További jelentős fejlemény a határokon átnyúló adatáramlás volumenének növekedése. A globális multinacionális nagyvállalatokon belüli feladatmegosztás gyakran azzal jár, hogy a vállalaton vagy vállaltcsoporton belüli adatáramlás is külföldi adattovábbítással jár együtt. Így sürgetővé vált a személyes adatok védelmének nagyobb harmonizációja.[53]

Az üzleti szférában lezajló folyamatokkal párhuzamosan a 80-as, 90-es évekre kiteljesedett az információs társadalom kialakulásával kapcsolatos elméleti diskurzus. Világossá vált, hogy a társadalmi változások fő mozgatórugója az információ felértékelődése és az információ felhasználásának hatékonysága lett. Ugyanakkor hangsúlyosan megjelennek a magánszférát féltő gondolatok is.[54]

A 90-es évek elejétől kezdődően az információs társadalom kiépítése az Európai Unió kiemelt politikai programjává vált. Mérföldkőnek tekinthető e területen az 1994-es Bangemann jelentés,[55] amelyet számos, az információs társadalom kialakítását célul tűző stratégiai dokumentum követett. E dokumentumokban rendre megjelenik az a kettősség, miszerint egyik oldalról biztosítani kell az adatok szabad áramlását, mivel az az információs társadalom fejlődésének motorja, másrészről azonban garantálni kell a magánszféra megfelelő védelmét is. Később, az internethasználat terjedésével, a megfelelő szintű adatvédelem – a fogyasztóvédelmi szabályok erősítése mellett – az online szolgáltatásokba vetett bizalom (trust) megteremtésének egyik fontos eszközévé, és így az információs társadalom kialakításának egyik lényegi szabályozási kérdésévé vált.[56]

### 1.3.2 A második generációs szabályozás főbb jellemzői

Az előző fejezetben bemutatott tendenciákkal összhangban az adatvédelmi jogalkotás is jelentős változáson ment keresztül. E változások elméleti előzményét a Német Szövetségi Alkotmánybíróság nagyhatású, ún. népszámlálás-ítéletében megfogalmazott információs önrendelkezési jog jelentette, amely „biztosítja az egyénnek azt a jogot, hogy alapvetően maga döntsön személyes adatainak kiszolgálásáról és felhasználásáról."[57] Az információs önrendelkezési jog elve áthatotta aztán nemcsak a német, de számos európai ország szabályozási rendszerét is.[58]

A második generációs szabályozás egyik fő jellemzője tehát, hogy a technológia-specifikus megközelítés helyett (amelyet a technológia gyors fejlődése reménytelenné tett) az érintettet széleskörű rendelkezési joggal ruházta fel az adatkezelés egész folyamatára nézve.[59]

Emellett széles körben elfogadottá vált, hogy – a „Kis Testvérek" megjelenésének köszönhetően – a szabályozás hatályát az üzleti szféra adatkezelőire is ki kell terjeszteni. Ugyancsak jellemző tendencia, hogy az elektronikus eszközökkel végzett adatkezelések mellett a manuális, papír alapú adatkezeléseket is rendre bevonta a jogalkotó a szabályozás hatálya alá.[60]

Az adatkezelések számának drasztikus növekedése, és az államigazgatás egyes szektorai esetén a jogalap biztosítása érdekében egyre nagyobb teret nyert az adatvédelem szektorális szabályozása. Ugyanakkor – ugyanezen okokból – a nyilvántartásba vételi szabályok kivételek sokaságával lazultak.[61]

Végül a szabályozás – több-kevesebb sikerrel – reagált a nemzetközi adattovábbításokból eredő problémákra is. Már a 80-as évek elejére megszülettek az első nemzetközi jogi dokumentumok, amelyek egyértelműen a határokon átnyúló adatáramlás növekvő jelentőségét mutatják. Az 1980-ban elfogadott, a magánélet védelméről és a személyes adatok határokon átívelő áramlásáról szóló OECD irányelveknek ugyan nincs kötelező ereje, de számos fontos alapvető elvet tartalmaz, és mivel részese az Egyesült Államok is,[62] az abban foglaltak Európa és az Egyesült Államok közötti közös nevezőnek tekinthetők.[63] Az Európa Tanács 1981-es adatvédelmi egyezménye[64] korának legjelentősebb adatvédelmi dokumentuma, és egészen 1995-ig, az EU adatvédelmi irányelvének elfogadásáig az egyetlen kötelező erejű, nemzetközi szintű jogforrás.[65]

53 See Burkert, Herbert im. 51. 53.
54 Többek között Masuda, Yoneji: Az információs társadalom. OMIKK, Budapest, 1988 102-110., Lussato, Bruno: Az informatikai kihívás, OMIKK, Budapest, 1989 152-157.
55 Balogh Zsolt György im. 159.
56 Az EU információstársadalom-politikájával kapcsolatban ld. http://europa.eu/legislation_summaries/information_society/index_hu.htm
57 Könyves-Tóth Pál – Székely Iván: Informatika – Jog – Közigazgatás. Nemzetközi dokumentumok I. InfoFilia, Budapest, 1991 p, 6.1.
58 Rouvroy, Antoinette – Poullet, Yves: The Right to Informational Self-Determination and the Value of Self-Deployment: Reassessing the Importance of Privacy for Democracy. in.: Serge Gutwirth et al. (eds.): Reinventing Data Protection? Springer, 2010 45-76. 45.
59 Mayer-Schönberger gondolatait idézi és elemzi Jóri András im. 27.
60 Hegedűs Bulcsú im. 135-136.
61 Jóri András im. 41.
62 Burkert, Herbert im. 52.
63 Jóri András im. 28-29. Az OECD irányelveket részletesen bemutatja továbbá Majtényi László: Az információs szabadságok, 95-96., Hegedűs Bulcsú im. 146-148.
64 Az egyének védelméről a személyes adatok gépi feldolgozása során, Strasbourgban, 1981. január 28. napján kelt Egyezmény
65 Az egyezményről ld. részletesen pl. Balogh Zsolt György im. 190-199., Jóri András im. 29-30., Hegedűs Bulcsú im. 148-150.

Az Európai Parlament és az Európai Unió Tanácsa – hosszas jogalkotási folyamat eredményeként – 1995-ben fogadta el az EU adatvédelmi irányelvét.[66] Az irányelv a második generációs szabályozás tipikus dokumentuma, amelynek implementálása minden EU tagállam számára kötelező volt, így e dokumentum megteremtette a harmonizált európai adatvédelmi szabályrendszer alapjait.

## 1.4 A harmadik generációs adatvédelmi szabályozás kialakulása és jellemzői

Az elmúlt 10-15 évben a technológiai és társadalmi háttér változása újra és újra kihívás elé állította és állítja a 90-es évek végére nagyjából kialakult adatvédelmi szabályozást. E tendenciák részletes feltérképezése, egy új generációs szabályozás elemeinek átfogó vizsgálata, valamint az ezek közötti okozati összefüggések feltárása további jelentős kutatást igényel, így e tanulmányban csupán arra vállalkozunk, hogy – mintegy felsorolásszerűen – azonosítsunk néhány fontosabb kérdéskört mind a technológiai és társadalmi változások, mind a várható új szabályozás főbb tendenciái közül.

### 1.4.1 Technológiai és társadalmi háttér

A technológiai-társadalmi háttér bemutatása kapcsán mindenekelőtt ki kell emelni az internetes szolgáltatások terén bekövetkezett változásokat, egyrészt az internethasználat tömeges elterjedését, másrészt a kétezres évek közepétől kezdődően a web 2.0-es szolgáltatások folyamatos elterjedését. Ezek egyik lényegi jellemzője, hogy drasztikusan megváltozik a tartalomszolgáltatás jellege, és előtérbe kerülnek a felhasználók által generált tartalmak,[67] legyen az akár egy személyes profiloldal egy közösségi oldalon, egy blogbejegyzés, vagy kép és videó megosztása. Amennyiben ezek a tartalmak más érintettek személyes adatainak nyilvánosságra hozatalával járnak, ez azt is eredményezheti, hogy felhasználók tömege kerül adatkezelői pozícióba,[68] és lesz így az adatvédelmi szabályozás kötelezettje.

A felhasználók által készített és közzétett tartalmaknak köszönhetően az online adatmennyiség korábban elképzelhetetlen mértékben bővül. A technológiai szaksajtó a „Big Data" kifejezéstől hangos, amely az adatvédelmi szakirodalomban is megjelent. A hatalmas és gyorsuló ütemben bővülő adatmennyiség hasznosítása komoly üzleti lehetőségeket rejt, így középpontba került az adatbányászati technológiák fejlesztése is. E jelenség magánszférára gyakorolt hatása nem megkerülhető az adatvédelemről szóló szakmai viták során.

További fontos tendencia a profilozás technikáinak fejlődése,[69] az egyre kifinomultabb direktmarketing-eszközök (pl. viselkedés alapú marketing) alkalmazása, a mobileszközök és ezzel összefüggésben a helymeghatározáson alapuló szolgáltatások nagyfokú elterjedése; utóbbi lényegében a valósidejű tartózkodási hely különböző adatkezelőnek történő megadásával jár együtt. Nagymértékben változik a személyes adatok tárolásának módja is. A felhőalapú szolgáltatások terjedésével az érintettek (sőt, sokszor az adatkezelők is) minden korábbinál jobban elvesztik az adataik feletti „fizikai" ellenőrzés lehetőségét.[70]

Érdemes figyelembe venni azt is, hogy az új technológiák és szolgáltatások új felhasználói viselkedésmintákkal párosulnak. A felhasználók új generációjának a magánszférával kapcsolatos attitűdje eltér a korábbi generációiétól.[71]

### 1.4.2 A harmadik generációs szabályozás néhány jellemzője

A fenti tendenciák hatása az adatvédelmi szabályozásban is meg kell, hogy jelenjen. Álláspontunk szerint olyan új generációs adatvédelmi szabályozás szükséges, amely alapvető jellemzőiben tér el a hatályos jogi környezettől – legalább annyiban, mint amennyiben a második generációs szabályok a korai adatvédelmi szabályozástól.

Amint azt említettük, folyamatban van az adatvédelmi szabályozás európai szintű reformja. A Bizottság által kiadott Rendelettervezet véleményünk szerint olyan új tendenciákat vetít előre, amelyek alapján egyértelműen új generációs szabályozásnak tekinthetjük.

Az alábbiakban összefoglalunk néhány jellemző szabályozási irányt.

*1)* A szabályozás súlypontja az érintettek jogai felől az adatkezelők kötelezettségei felé tolódik el. Úgy tűnik, hogy az információs önrendelkezési jog egyéni jogérvényesítést és az érintett tudatosságát feltételező koncepciója mellett/helyett fokozottan előtérbe kerül az adatkezelők kötelezettségeit és felelősségét (elszámoltathatóságát) kidomborító megközelítés.

66 95/46/EK irányelv

67 User Generated Content

68 Van Alsenoy, Brendan - Ballet, Joris – Kuczerawy Aleksandra – Dumortier Jos: Social networks and web 2.0: are users also bound by data protection regulations? Identity in the Information Society, 2009/1, doi: 10.1007/s12394-009-0017-3 [2013.08.15.] 70.

69 A profilozás széleskörű alkalmazási lehetőségeiről ld. Hildebrandt, Mireille – Gutwirth Serge (eds.): Profiling the European Citizen. Cross Disciplinary Perspectives. Springer, 2010

70 Tene, Omer: Privacy: The new generations, International Data Privacy Law 20011/1. doi: 10.1093/idpl/ipq003 [2013.07.11.] 16-20.

71 Tene,Omer im. 15, 21, 23. A felhasználói hozzáállás részletesebb vizsgálatához további kutatások szükségesek – amely elsősorban az elmúlt években készült közvélemény-kutatások eredményeinek feldolgozását igényli.

A Bizottság által kibocsátott rendelettervezet az adatkezelők mérete és tevékenysége alapján differenciáltan, de széles adatkezelői kört érintően az adatkezelők feladatává teszi a megfelelő politikák (eljárásrendek, szabályzatok) elfogadását, adatkezelési dokumentáció vezetését, adatbiztonsági intézkedések megtételét, adatvédelmi hatásvizsgálat lefolytatását, és egyes esetekben belső adatvédelmi felelős kijelölését.[72]

A szabályozás ugyan elvileg az érintett jogainak erősítését is célul tűzi ki, de véleményünk szerint ezek vagy jóval kisebb előrelépések, mint amelyeket a kötelezettségek területén ír elő a tervezet, vagy olyan új jogokat nevesít, amelynek biztosítása jelentős erőfeszítést igényel az adatkezelők részéről (pl. az adathordozhatósághoz való jog).

Apró változtatásnak tűnik, de jelentős koncepcionális váltást eredményezhet a Rendelettervezet azon szakasza, amely szerint a hozzájárulás nem teremt jogalapot az adatkezelésre, ha jelentős egyenlőtlenség áll fenn az érintett és az adatkezelő helyzete között.[73] Megjegyezzük, hogy az információs önrendelkezési jogon alapuló adatvédelmi szabályozással kapcsolatos aggályok már jó ideje megjelennek a jogirodalomban. Mayer-Schönberger már 1997-ben arra hívja fel a figyelmet, hogy az információs önrendelkezési jogon alapuló adatvédelem „fogatlan papírtigrisnek", a „felső középosztály játékszerének bizonyul," mivel az érintettek többsége rutinszerűen megadja a jobb gazdasági és alkupozícióban lévő gazdálkodó szervezetnek az adatkezeléshez való hozzájárulást.[74]

Összességében tehát a Rendelettervezet reagálni igyekszik ezekre a felvetésekre, és – hasonlóan például a fogyasztóvédelmi szabályozáshoz – az érintett passzivitása, alacsony adatvédelmi tudatossága mellett is biztosítani kívánja (talán tudja is), a magánszféra-védelem elfogadható szintjét.

*2)* Egy másik fontos jellemző a szabályozási terhek differenciálása az adatkezelők valamely jellemzője alapján. A Rendelettervezet az egyes, részletesen szabályozott kötelezettségeket csak bizonyos adatkezelők számára írja elő. A tervezet egyrészt az adatkezelés jellege (fő- vagy járulékos tevékenység) és kockázata alapján, másrészt az adatkezelő mérete alapján differenciál.

*3)* A szabályozás (újra) célul tűzi ki a technológia szabályozását, formálását. A jogalkotó felismerte ugyanis, hogy a technológia közvetlen hatással van a személyes adatok védelmére és az adatbiztonságra vonatkozó jogi követelményekre.[75] Hiába van pl. egy szervezetnél az adatvédelmi jogszabályoknak egyébként megfelelő belső szabályzat arra, hogy bizonyos adatokhoz csak meghatározott szervezeti egységek férhetnek hozzá, ha ezt nem támogatják az iktatórendszer jogosultsági beállításai, esetleg nem is lehet megadni jogosultsági korlátozásokat, akkor e szabály a gyakorlatban nem fog érvényesülni.

A technológia azonban kifejezetten segítheti is a magánszféra védelmét; a privátszférát erősítő technológiák[76] (PET) létrejöttének egyik oka, hogy a szabályozás, önszabályozás, és a jogalkalmazás sem tudnak elegendő védelmet nyújtani a felhasználóknak a tömegesen előforduló jogellenes adatkezelési gyakorlat, és az egyre újabb műszaki megoldások ellen.[77]

A fentiekre tekintettel – a PET koncepcióját is magába olvasztva – egyre többször jelenik meg jogszabályi követelményként is az ún. „Privacy by design" elve, amely lényegében azt jelenti, hogy valamely új technológia illetve eljárás fejlesztése és/vagy bevezetése során már a tervezési szakasztól kezdve (egészen a tényleges alkalmazásig) figyelemmel kell lenni az adatvédelemre vonatkozó jogi, és az adatbiztonságra vonatkozó jogi és technikai követelményekre, valamint műszakilag biztosítani kell az azoknak való megfelelést. Szintén új szabályként jelenik meg az új európai szabályozásban bizonyos szervezetek számára az adatvédelmi hatásvizsgálat (privacy impact assessment) követelménye, amely valamely intézkedés egyének magánszférájára gyakorolt hatásának elemzését jelenti.[78]

*4)* A fenti jellemzőkből következik az intézményi belső szabályozás szerepének erősödése. A Rendelettervezet jelen formában való elfogadásának véleményünk szerint egyértelműen az a várható hatása, hogy az egyes, különösen a nagyobb, vagy a személyes adatok kezelését elsődleges tevékenységként végző adatkezelők a korábbinál lényegesen nagyobb hangsúlyt kell, hogy fektessenek az adatvédelmet érintő folyamataik és azok várható hatásainak elemzésére, majd e folyamatok megtervezésére, dokumentálására és (belső) felügyeletére.

72 Rendelettervezet, IV. fejezet
73 Rendelettervezet, 7. cikk (4) bekezdés
74 Mayer-Schönberger gondolatatit idézi és elemzi Jóri András im. 37. A második generációs szabályozás válságáról ld. részletesen Jóri András im. 36-42.
75 E gondolatnak az első jogszabályi megfogalmazása már 1997-ben megjelent Németországban. Ld. Erről Jóri András im. 65.
76 Privacy Enhancing Technology (PET)
77 Székely Iván: Privátszférát erősítő technológiák. In: Információs Társadalom, 2008. VIII. évf. 1. szám, http://pet-portal.eu/files/oldfiles/articles/2008/02/InfTars_PET.pdf (2013.06.27.), 22.
78 Rendelettervezet, 23. és 33. cikk

# II. Az adatvédelem hazai szabályozása

## 1. Alkotmányos háttér

1989-ben a jogállami forradalom keretében a személyes adatok védelméhez való jog (és a közérdekű adatok nyilvánossága is) bekerült az Alkotmányba.[79]

Az Alkotmánybíróság a személyes adatok védelméhez való jogot – német minta alapján – nem hagyományos védelmi jogként, hanem annak aktív oldalát is figyelembe véve információs önrendelkezési jogként értelmezte:[80] „a személyes adatok védelméhez való jognak eszerint az a tartalma, hogy mindenki maga rendelkezik személyes adatainak feltárásáról és felhasználásáról. Személyes adatot felvenni és felhasználni tehát általában csakis az érintett beleegyezésével szabad; mindenki számára követhetővé és ellenőrizhetővé kell tenni az adatfeldolgozás egész útját, vagyis mindenkinek joga van tudni, ki, hol, mikor, milyen célra használja fel az ő személyes adatát. Kivételesen törvény elrendelheti személyes adat kötelező kiszolgáltatását, és előírhatja a felhasználás módját is. Az ilyen törvény korlátozza az információs önrendelkezés alapvető jogát,"[81] és a korlátozás csak akkor alkotmányos, ha megfelel az alapjogok korlátozására vonatkozó általános szabályoknak.

A személyes adatok védelmének törvényi szintű szabályozására először 1992-ben került sor. A magyar jogalkotó az információs önrendelkezési jogot és az információszabadságot ugyanabban a törvényben, a személyes adatok védelméről és a közérdekű adatok nyilvánosságáról szóló, természetesen számos alkalommal módosított 1992. évi LXIII. törvényben szabályozta.

Az 2012. január 1-től hatályos új Alaptörvény – követve a jogirodalomban kialakult többségi álláspontot – közös bekezdésben említi az információs alapjogokat: a személyes adatok védelmét és a közérdekű adatok nyilvánosságát. Az Alaptörvény nevesíti azt is, hogy e két alapvető jog érvényesülését sarkalatos törvénnyel létrehozott független hatóság ellenőrzi.[82]

2011. július 11-én az Országgyűlés elfogadta az információs önrendelkezési jogról és az információszabadságról szóló 2011. évi CXII. törvényt (Infotv. vagy adatvédelmi törvény), amely 2012. január 1-től felváltotta az 1992-es szabályozást. Az új adatvédelmi törvény – bár nagymértékben támaszkodik a korábbi szabályozásra, és számtalan rendelkezést szó szerint átvesz a hatályos törvényekből – számos ponton módosítja is a korábbi szabályozást, és teljesen átalakítja az adatvédelem felügyeleti rendszerét.

Az adatvédelem szabályozása nem oldható meg egyetlen törvényben. Az Infotv. a minden adatkezelés során figyelembe veendő általános követelményeket és garanciákat határozza meg, amiket azonban az adott adatkezelés sajátosságaira koncentráló, az általános szabályokat konkretizáló ágazati adatvédelmi jogszabályok egészítenek ki.

## 2. Alapfogalmak

### 2.1 Személyes adat, különleges adat

Az Infotv. a személyes adat fogalmát igen szélesen határozza meg, személyes adat az érintettel kapcsolatba hozható adat – különösen az érintett neve, azonosító jele, valamint egy vagy több fizikai, fiziológiai, mentális, gazdasági, kulturális vagy szociális azonosságára jellemző ismeret –, valamint az adatból levonható, az érintettre vonatkozó következtetés.

A rendkívül tág fogalomkörön belül a jogirodalomban található csoportosítás alapján a két legalapvetőbb kategóriát az azonosító és a leíró adatok csoportja képezi.

*1)* Az azonosító adatok az adatkezeléssel érintett személy egyediesítését, másoktól való megkülönböztetését szolgálják. Erre a célra természetes és mesterséges azonosító adatok használhatók fel. Természetes azonosítók különösen a név, a születés helye és időpontja, az anya neve, valamint a lakcímadatok (ezekből általában többet kell alkalmazni az azonosításhoz). A mesterséges azonosítók valamilyen matematikai illetve statisztikai eljárással generált kódok (többnyire számok vagy számok és betűk kombinációja); mesterséges azonosító például a személyi igazolvány száma, az útlevélszám, a vezetői engedély száma, az adóazonosító jel, a társadalombiztosítási azonosító jel (TAJ). A mesterséges azonosítóból egy is elég a személy egyediesítéséhez.

79 1949. XX. törvény a Magyar Köztársaság Alkotmánya, 59. § és 61. §

80 Részletesen ld. Majtényi, László: Az információs szabadságok, 168-175., Balogh Zsolt György im. 175-176.

81 15/1991. (IV. 13.) AB hat.

82 Magyarország Alaptörvénye, VI. cikk (2)-(3) bekezdés

*2)* A leíró adatok az adatkezelés célja szerint releváns személyes adatok. Az azonosítókon kívül minden az adatkezelésbe bevont személyes adat e kategóriába tartozik. A leíró adatok az érintett valamilyen tulajdonságát, jellemzőjét, különböző viszonyait fejezik ki.[83]

A személyes adat fogalma egy másik fogalommal, az érintett fogalmával lesz teljes. Az Infotv. alapján az érintett bármely meghatározott, személyes adat alapján azonosított vagy – közvetlenül vagy közvetve – azonosítható természetes személy. A törvény szerint személyes adata csak természetes személyeknek van; a jogi személyekre és más szervezetekre az adatvédelmi garanciák nem terjednek ki.

A törvény szigorúbb adatkezelési feltételeket határoz meg a személyes adatoknak arra a körére, amelyekkel való visszaélés súlyosabb következményekkel járhat (szenzitív vagy érzékeny adat). Az Infotv. ezeket az adatokat különleges adatnak nevezi. Ezek közé tartozik – a törvény zárt felsorolása szerint – egyrészt a faji eredetre, a nemzeti és etnikai kisebbséghez tartozásra, a politikai véleményre vagy pártállásra, a vallásos vagy más világnézeti meggyőződésre, az érdek-képviseleti szervezeti tagságra, másrészt az egészségi állapotra, a kóros szenvedélyre, a szexuális életre vonatkozó adat, valamint a bűnügyi személyes adat.[84]

## 2.2 Adatkezelés, adatfeldolgozás, adatkezelő, adatfeldolgozó

Az adatkezelés az adatokon végzett bármely művelet vagy a műveletek összessége, függetlenül az alkalmazott eljárástól, azaz a törvényt mind az automatizált, mind a manuális módon végzett adatkezelésekre alkalmazni kell. Adatkezelés – a törvény példálózó felsorolásában – az adatok gyűjtése, felvétele, rögzítése, rendszerezése, tárolása, megváltoztatása, felhasználása, továbbítása, nyilvánosságra hozatala, összehangolása vagy összekapcsolása, zárolása, törlése és megsemmisítése, valamint az adatok további felhasználásának megakadályozása, a fénykép-, hang- vagy képfelvétel készítése.[85]

Az a természetes személy vagy jogi személy, illetve jogi személyiséggel nem rendelkező szervezet, aki vagy amely az adatok kezelésének célját meghatározza, az adatkezelésre vonatkozó döntéseket meghozza, és e döntéseket – maga vagy egy adatfeldolgozó megbízásával – végrehajtja, az adatkezelő.[86]

Az adatkezelési műveletekhez kapcsolódó technikai feladatok – az alkalmazott módszertől és eszköztől, valamint az alkalmazás helyétől független – elvégzése az adatvédelmi törvény szerint adatfeldolgozásnak minősül. Az a természetes vagy jogi személy, illetve jogi személyiséggel nem rendelkező szervezet, aki vagy amely az adatkezelő megbízásából személyes adatok feldolgozását végzi, az adatfeldolgozó. Az adatkezelő és az adatfeldolgozó közötti megbízási jogviszonyból következően az adatfeldolgozó az adatkezelő utasításainak megfelelően végzi a tevékenységét, ő maga az adatkezelést érintő érdemi döntést nem hozhat. Felelőssége a saját tevékenységi körén belül a személyes adatok feldolgozásáért, megváltoztatásáért, törléséért, továbbításáért és nyilvánosságra hozataláért áll fenn. Az általa az érintettnek okozott kárért az érintettel szemben az adatkezelő felel.[87]

Adatfeldolgozói jogviszony a közigazgatás területén jellemzően olyankor jön létre, ha az adatkezelő szerv valamely olyan tevékenységét szervezi ki, amelynek keretében a szolgáltatást nyújtó szervezetnek személyes adatokhoz kell hozzáférnie és azon valamilyen műveletet végeznie.

## 2.3 Az Infotv. hatálya

Az Infotv. rendelkezéseinek hatálya a Magyarország területén folytatott minden olyan adatkezelésre és adatfeldolgozásra kiterjed, amely természetes személy adataira, valamint közérdekű adatra vagy közérdekből nyilvános adatra vonatkozik. A törvény területi hatálya tehát Magyarországra terjed ki.

A törvényt az adatkezelés módjától függetlenül a teljesen vagy részben automatizált eszközzel, valamint a manuális módon végzett adatkezelésre és adatfeldolgozásra is alkalmazni kell.

Végül fontos kivétel, hogy nem kell alkalmazni az adatvédelmi szabályokat a természetes személynek a kizárólag saját személyes céljait szolgáló adatkezeléseire.[88] E kivétel nélkül előállhatna az a nyilvánvalóan értelmetlen helyzet, hogy egy magánszemély mobiltelefonos telefonkönyve (amelyben a személy mások személyes adatait tárolja, ami tehát adatkezelésnek minősül) is az adatvédelmi szabályozás hatálya alá esne. Megjegyzendő ugyanakkor, hogy amennyiben azonban e telefonszámokat a személy a saját személyes célján kívüli célból, például az újonnan indított vállalkozásának népszerűsítésére használja, úgy ezen adatkezelés már az Infotv. hatálya alá kerül.

83 Balogh Zsolt György im. 170-171.
84 Infotv. 3. § 3. pont
85 Infotv. 3. § 10. pont
86 Infotv. 3. § 9. pont
87 Infotv. 3. § 17-18. pont, 10. §
88 Infotv. 2. §

# 3. Az adatkezelés jogalapja

A 80-as évektől kezdődően az Európai adatvédelmi szabályozás egyik legfontosabb jellemzőjévé vált, hogy a személyes adatok kezelésére csak meghatározott, jogszabályban tételesen felsorolt jogalapok megléte esetén kerülhet sor. Az adatkezelőnek tehát minden esetben igazolni kell tudnia az adatkezelés jogalapját.

## 3.1 Az érintett hozzájárulása

Az információs önrendelkezési jogból következően az adatvédelem rendszerében az első és legfontosabb jogalap az érintett adatkezeléshez történő hozzájárulása.

A törvény meghatározása alapján a hozzájárulás az érintettnek az a nyilatkozata, amellyel félreérthetetlen beleegyezését adja a rá vonatkozó személyes adatok – teljes körű vagy egyes műveletekre kiterjedő – kezeléséhez. Az érintett nyilatkozata akkor tekinthető érvényes hozzájárulásnak, ha az önkéntes és határozott, továbbá megfelelő tájékoztatáson alapul. Az érintettet ezért már az adat felvétele előtt tájékoztatni kell az adatkezelés legfontosabb körülményeiről. A tájékoztatásnak ki kell terjednie az adatkezelés önkéntes vagy kötelező jellegére, az adatkezelés céljára és jogalapjára, az adatkezelésre és az adatfeldolgozásra jogosult személyére, az adatkezelés időtartamára, az adatokat megismerők személyére, illetve az érintett adatkezeléssel kapcsolatos jogaira és jogorvoslati lehetőségeire.[89]

A hozzájárulás általában nincs alakiságokhoz kötve, azaz szóban, sőt ráutaló magatartással is megadható (az adatkezelés „eltűrése" azonban nem minősül hozzájárulásnak, mert annak kifejezettnek kell lennie). A törvény szigorúbb alakiságot követel meg a különleges adatok kezelése esetén: a különleges adatok kezeléséhez történő hozzájárulásnak írásbelinek kell lennie.

A törvény megállapít két hozzájárulási vélelmet is. Egyrészt megadottnak kell tekinteni az adatkezeléshez való hozzájárulást az érintett közszereplése során általa közölt vagy a nyilvánosságra hozatal céljából általa átadott adatok tekintetében. Másrészt vélelmezni kell a hozzájárulást az érintett kérelmére, kezdeményezésére indult bírósági vagy hatósági eljárásban az eljárás lefolytatásához szükséges személyes adatok tekintetében, az érintett kérelmére indult más ügyben az általa megadott személyes adatok tekintetében. E vélelmek kiterjednek a különleges adatok esetére is.

## 3.2 Jogszabályon alapuló adatkezelés

Az érintett hozzájárulásán kívül személyes adat kezelését közérdekből törvény, valamint törvény felhatalmazása alapján kiadott helyi önkormányzati rendelet is előírhatja (kötelező adatkezelés). A kötelező adatkezelés célját és egyéb feltételeit az adatkezelést elrendelő jogszabály határozza meg. A korábban említett ágazati adatkezelési szabályok sok esetben éppen ilyen, adatkezelést elrendelő jogszabályoknak tekinthetők, amelyek gyakran nem önálló törvényként, hanem az adott szektort szabályozó törvény részeként jelennek meg.

Megjegyzendő, hogy a közigazgatás adatkezeléseinek döntő többsége törvényen alapuló adatkezelés. Akár az adóhatóság, akár a bűnüldöző szervek helyzetére gondolva könnyen belátható, hogy az érintett önkéntességén alapuló adatkezelés veszélyeztetné e szervek munkáját. A jogszabályon alapuló adatkezelés tehát az érintett akarata ellenére is jogszerű lehet.

Fel kell hívni ugyanakkor arra a figyelmet, hogy a törvényen alapuló adatkezelés csak az ott meghatározott célra, adatkörre, időtartamra terjedhet ki. Az ezeken kívül eső adatkezelések esetén más jogalapra van szükség (ami lehet az érintett kifejezett vagy a fenti szabályok alapján vélelmezett hozzájárulása vagy valamely más jogszabály is).

## 3.3 Érdekmérlegelésen és jogi kötelezettség teljesítésén alapuló adatkezelés

Az új Infotv. talán legjelentősebb újdonsága az adatkezelési jogalapok bővítése. A korábbi adatvédelmi törvény kizárólag két esetben tette lehetővé a személyes adatok kezelését: ha ehhez az érintett hozzájárult, vagy ha ezt törvény, illetve törvény felhatalmazása alapján, az abban meghatározott körben helyi önkormányzat rendelete elrendelte.

Az információs önrendelkezési jog ilyen következetes érvényesítése Európában is egyedülálló, ugyanakkor e szabályozás a jogalkalmazói gyakorlatban csak meglehetősen rugalmas jogértelmezéssel volt hozzáilleszthető a felmerülő problémákhoz, és az európai közösségi joggal való összhangja is vitatható volt: az EU adatvédelmi irányelvének 7. cikkének f) pontja szerint személyes adatok kezelhetők többek között abban az esetben, ha az adatkezelés az adatkezelő, vagy az adatokat megkapó harmadik fél, vagy felek jogszerű érdekének érvényesítéséhez szükséges, kivéve, ha ezeknél az érdekeknél magasabb rendű az érintettnek a magánélet tiszteletben tartásához való joga. Az érdekmérlegelés jelentősen kitágítja a jogszerű adatkezelések körét, és egyúttal szükségszerűen bizonytalanabbá is teszi azok határait.[90]

89 Infotv. 3. § 7. pont, 20. §

90 Polyák Gábor – Szőke Gergely László: Elszalasztott lehetőség? Az új adatvédelmi törvény főbb rendelkezései, In.: Drinóczi Tímea (szerk.): Magyarország új alkotmányossága, Pécsi Tudományegyetem, Állam- és Jogtudományi Kar, 2011, 160.

Az Infotv. alapján személyes adat kezelhető akkor is, ha az érintett hozzájárulásának beszerzése lehetetlen vagy aránytalan költséggel járna,[91] és a személyes adat kezelése

*1)* az adatkezelőre vonatkozó jogi kötelezettség teljesítése céljából szükséges, vagy

*2)* az adatkezelő vagy harmadik személy jogos érdekének érvényesítése céljából szükséges, és ezen érdek érvényesítése a személyes adatok védelméhez fűződő jog korlátozásával arányban áll.[92]

Az érdekmérlegelésen alapuló jogalap megjelenése indokolt mértékű rugalmasságot hoz a szabályozásba, és világos helyzetet teremt számos, jelenleg jogsértő, de jogkövetkezmény nélkül maradó adatkezelés számára. Ilyen jogsértések állhattak elő többek között a munkáltató adatkezelési gyakorlatában, amikor pontosan meghatározott jogalap nélkül ellenőrizte a munkavállalók tevékenységét, vagy a munkáltatók (akár a közigazgatási szervek, intézmények, akár magánvállalkozások) a szervezetnél dolgozók részére nyújtottak hírközlési szolgáltatásokhoz kapcsolódó szolgáltatásokat (pl.: e-mail szolgáltatást tárhellyel), vagy az oknyomozó újságíró tevékenységében.[93]

Meg kell ugyanakkor jegyezni, hogy a NAIH joggyakorlata alapján nincs helye az érdekmérlegelésen alapuló adatkezeléseknek azokon a területeken, ahol az adatkezelés részleteit törvény szabályozza, mivel itt a jogalkotó már mérlegelte a különböző érdekek egyensúlyát. Tekintettel arra, hogy a közigazgatási adatkezelések döntő többsége törvényen alapul, az érdekmérlegelési jogalap alkalmazása a közszféra adatkezeléseiben csak kivételes esetekben merülhet fel.

## 4. A célhoz kötöttség követelménye és az adatok minősége

A személyes adatok kezelésének legfontosabb garanciája, hogy arra minden esetben pontosan meghatározott, jogszerű cél teljesítése érdekében kerüljön sor (célhoz kötöttség). „A célhozkötöttségből következik, hogy a meghatározott cél nélküli, »készletre«, előre nem meghatározott jövőbeni felhasználásra való adatgyűjtés és -tárolás alkotmányellenes."[94]

A célhoz kötöttség garanciáját a törvény összetett követelményként fogalmazza meg:

- személyes adatot kezelni csak meghatározott célból,[95] jog gyakorlása és kötelezettség teljesítése érdekében lehet;
- a célhoz kötöttségnek az adatkezelés minden szakaszában teljesülnie kell, azaz az adatkezelésnek minden szakaszában meg kell felelnie az adatkezelési célnak;
- csak az adatkezelés céljának megvalósulásához elengedhetetlen és a cél elérésére alkalmas személyes adat kezelésére kerülhet sor;
- az adatkezelés nem haladhatja meg a cél megvalósulásához szükséges mértéket és időtartamot;
- ha az adatkezelés célja megszűnt, akkor a személyes adatot törölni kell;
- a célhoz kötöttség technikai garanciájaként az adattárolás módjának alkalmasnak kell lennie arra, hogy az érintettet csak a tárolás céljához szükséges ideig lehessen azonosítani.[96]

Fontos hangsúlyozni, hogy az adatvédelmi biztos és a Nemzeti Adatvédelmi és Információszabadság Hatóság (NAIH) következetes joggyakorlata alapján a célhoz kötöttség követelményét a már nyilvánosságra hozott személyes adatok tekintetében is alkalmazni kell, azaz a nyilvánosságra hozatal céljától eltérő más célból azok újrafelhasználása új adatkezelésnek minősül, amely csak megfelelő jogalappal lehetséges.

Az adatvédelmi törvény a kezelt adatok minőségére vonatkozó követelményeket is meghatároz. Eszerint az adatok felvétele és kezelése tisztességes és törvényes kell, hogy legyen. A tisztességes adatkezelés követelménye a törvényesség mércéjénél szigorúbb, pontos tartalma azonban általános érvénnyel nem határozható meg. Az adatminőség követelményeként a törvény előírja továbbá, hogy a kezelt személyes adatok legyenek pontosak, teljesek és, ha szükséges, időszerűek.[97]

91 Megjegyezzük, hogy az Infotv. ezen kitétele az érdekmérlegelés jogalapját feltételekhez köti, ami ellentétes az EU adatvédelmi irányelvével. (Forrás)

92 Infotv. 6. § (1)

93 Polyák – Szőke im. 160.

94 15/1991. (IV. 13.) AB hat.

95 A törvény példálózó felsorolást tartalmaz a lehetséges adatkezelési célokra vonatkozóan. A személyes adatot – akár az érintett hozzájárulásával, akár jogszabály alapján – különösen akkor lehet kezelni, ha ez közérdekű feladat vagy az adatkezelő törvényi kötelezettségének teljesítéséhez, az adatkezelő vagy az adatátvevő harmadik személy hivatalos feladatának gyakorlásához, az érintett létfontosságú érdekeinek védelméhez, az érintett és az adatkezelő között létrejött szerződés teljesítéséhez, az adatkezelő vagy harmadik személy jogos érdekének érvényesítéséhez, társadalmi szervezetek jogszerű működéséhez szükséges.

96 Gálik, Mihály – Polyák Gábor: Médiaszabályozás, KJK-Kerszöv, 2005, 220.

97 Infotv. 4. §

# 5. További szabályok az adatkezeléssel kapcsolatban

## 5.1 Adatvédelmi nyilvántartás

Az adatkezelési tevékenység az Infotv. alapján hatósági nyilvántartásba vételhez, mint adminisztratív feltételhez kötött tevékenység. A személyes adatok kezelésének nyilvántartásba vételét az adatkezelő – a kötelező adatkezelés kivételével az adatkezelés megkezdése előtt – kérelmezi a Hatóságnál. A kötelező adatkezelés, valamint a NAIH hallgatása kivételével az adatkezelés a nyilvántartásba vételt megelőzően nem kezdhető meg.

A Hatóság az adatkezelést a kérelem megérkezésétől számított nyolc napon belül nyilvántartásba veszi, ha a kérelem tartalmazza törvényben előírt adatokat. Ha a Hatóság a nyilvántartásba vétel iránti kérelmet határidőben nem bírálja el, az adatkezelő az adatkezelést a kérelemben foglaltak szerint megkezdheti.

Az adatkezelő személyes adatokra vonatkozó adatkezeléseiről, az érintettek tájékozódásának elősegítése érdekében a Hatóság hatósági nyilvántartást (adatvédelmi nyilvántartás) vezet, amely tartalmazza

*a)* az adatkezelés célját,
*b)* az adatkezelés jogalapját,
*c)* az érintettek körét,
*d)* az érintettekre vonatkozó adatok leírását,
*e)* az adatok forrását,
*f)* az adatok kezelésének időtartamát,
*g)* a továbbított adatok fajtáját, címzettjét és a továbbítás jogalapját, ideértve a harmadik országokba irányuló adattovábbításokat is,
*h)* az adatkezelő, valamint az adatfeldolgozó nevét és címét, a tényleges adatkezelés, illetve az adatfeldolgozás helyét és az adatfeldolgozónak az adatkezeléssel összefüggő tevékenységét,
*i)* az alkalmazott adatfeldolgozási technológia jellegét,
*j)* a belső adatvédelmi felelős alkalmazása esetén annak nevét és elérhetőségi adatait.

Az adatvédelmi nyilvántartás a nemzetbiztonsági szervek adatkezelései tekintetében a nemzetbiztonsági szerv nevét és címét, az adatkezelés célját és jogalapját tartalmazza.

Nem vezet adatvédelmi nyilvántartást a Hatóság arról az adatkezelésről, amely

*a)* az adatkezelővel munkaviszonyban, tagsági viszonyban, óvodai nevelésben való részvételre irányuló, tanulói vagy tanulószerződéses jogviszonyban, kollégiumi tagsági viszonyban vagy - a pénzügyi szervezetek, közüzemi szolgáltatók, elektronikus hírközlési szolgáltatók ügyfelei kivételével - ügyfélkapcsolatban álló személyek adataira vonatkozik;
*b)* egyház, vallásfelekezet, vallási közösség belső szabályai szerint történik;
*c)* az egészségügyi ellátásban kezelt személy betegségével, egészségi állapotával kapcsolatos személyes adatokra vonatkozik gyógykezelés vagy az egészség megőrzése, társadalombiztosítási igény érvényesítése céljából;
*d)* az érintett anyagi és egyéb szociális támogatása céljából nyilvántartott személyes adatokra vonatkozik;
*e)* a hatósági, az ügyészségi és a bírósági eljárás által érintett személyeknek az eljárás lefolytatásával kapcsolatos személyes adataira, vagy a büntetés-végrehajtás során a büntetés-végrehajtással összefüggésben kezelt személyes adatokra vonatkozik;
*f)* a hivatalos statisztika célját szolgáló személyes adatokat tartalmaz, feltéve, hogy - törvényben meghatározottak szerint - az adatok érintettel való kapcsolatának megállapítását véglegesen lehetetlenné teszik;
*g)* a médiaszolgáltatásokról és a tömegkommunikációról szóló törvény szerinti médiatartalom-szolgáltató olyan adatait tartalmazza, amelyek kizárólag saját tájékoztatási tevékenységét szolgálják;
*h)* a tudományos kutatás céljait szolgálja, ha az adatokat nem hozzák nyilvánosságra;
*i)* a levéltári őrizetbe vett iratokkal összefüggésben valósul meg.

A fenti szabályok alapján tehát a közigazgatási adatkezelések főszabályként mentesülnek a bejelentési kötelezettség alól.

## 5.2 A személyes adatok továbbítása külföldre

Személyes adat harmadik országban adatkezelést folytató adatkezelő részére akkor továbbítható, vagy harmadik országban adatfeldolgozást végző adatfeldolgozó részére akkor adhat át, ha

*1)* ahhoz az érintett kifejezetten hozzájárult, vagy
*2)* más jogalapon alapuló adatkezelések esetén a harmadik országban az átadott adatok kezelése, valamint feldolgozása során biztosított a személyes adatok megfelelő szintű védelme.

A személyes adatok megfelelő szintű védelme akkor biztosított, ha az Európai Unió kötelező jogi aktusa azt megállapítja, vagy a harmadik ország és Magyarország között az adatvédelem megfelelő szintjét garanciális szabályokkal biztosító nemzetközi szerződés van hatályban. A személyes adatok a nemzetközi jogsegélyről, az adóügyi információcseréről,

valamint a kettős adóztatás elkerüléséről szóló nemzetközi szerződés végrehajtása érdekében, a nemzetközi szerződésben meghatározott célból, feltételekkel és adatkörben az előző feltételek hiányában is továbbíthatók harmadik országba.

Az Infotv. rögzíti, hogy az EGT-államba[98] irányuló adattovábbítást úgy kell tekinteni, mintha Magyarország területén belüli adattovábbításra kerülne sor, a fenti, harmadik országba irányuló szabályok tehát az Európai Unión belüli adattovábbításra nem vonatkoznak.

### 5.3 Az adatbiztonságra vonatkozó szabályok

Az adatkezelés akkor lehet törvényes és tisztességes, illetve az adatvédelmi követelmények csak akkor teljesíthetők, ha az adatkezelés technikai és szervezeti háttere azt lehetővé teszi. A törvény ezért előírja az adatbiztonság feltételeinek megteremtését.

Az Infotv. kimondja, hogy az adatkezelő köteles az adatkezelési műveleteket úgy megtervezni és végrehajtani, hogy az e törvény és az adatkezelésre vonatkozó más szabályok alkalmazása során biztosítsa az érintettek magánszférájának védelmét. Ez tulajdonképpen a korábban már tárgyalt beépített adatvédelem (Privacy by Design) elvének törvénybe foglalása.

Emellett az adatkezelő, illetve tevékenységi körében az adatfeldolgozó köteles gondoskodni az adatok biztonságáról, köteles továbbá megtenni azokat a technikai és szervezési intézkedéseket és kialakítani azokat az eljárási szabályokat, amelyek e törvény, valamint az egyéb adat- és titokvédelmi szabályok érvényre juttatásához szükségesek. Az adatokat megfelelő intézkedésekkel védeni kell különösen a jogosulatlan hozzáférés, megváltoztatás, továbbítás, nyilvánosságra hozatal, törlés vagy megsemmisítés, valamint a véletlen megsemmisülés és sérülés, továbbá az alkalmazott technika megváltozásából fakadó hozzáférhetetlenné válás ellen. A személyes adatok automatizált feldolgozása során az adatkezelő és az adatfeldolgozó további intézkedésekkel biztosítja

- a jogosulatlan adatbevitel megakadályozását;
- az automatikus adatfeldolgozó rendszerek jogosulatlan személyek általi, adatátviteli berendezés segítségével történő használatának megakadályozását;
- annak ellenőrizhetőségét és megállapíthatóságát, hogy a személyes adatokat adatátviteli berendezés alkalmazásával mely szerveknek továbbították vagy továbbíthatják;
- annak ellenőrizhetőségét és megállapíthatóságát, hogy mely személyes adatokat, mikor és ki vitte be az automatikus adatfeldolgozó rendszerekbe;
- a telepített rendszerek üzemzavar esetén történő helyreállíthatóságát és
- azt, hogy az automatizált feldolgozás során fellépő hibákról jelentés készüljön.

További követelmény, hogy a különböző nyilvántartásokban elektronikusan kezelt adatállományok védelme érdekében megfelelő technikai megoldással biztosítani kell, hogy a nyilvántartásokban tárolt adatok – kivéve, ha azt törvény lehetővé teszi – közvetlenül ne legyenek összekapcsolhatók és az érintetthez rendelhetők.

A törvény elvi szinten rögzíti a technológia állásának figyelembevételét is, de a törvényben nincs külön utalás a kockázatarányos védelem elvére. Az Infotv. kimondja, hogy az adatkezelőnek és az adatfeldolgozónak az adatok biztonságát szolgáló intézkedések meghatározásakor és alkalmazásakor tekintettel kell lenni a technika mindenkori fejlettségére. Több lehetséges adatkezelési megoldás közül azt kell választani, amely a személyes adatok magasabb szintű védelmét biztosítja, kivéve, ha az aránytalan nehézséget jelentene az adatkezelőnek.

A – ténylegesen igen bizonytalan tartalmú – adatbiztonsági szabályok megsértése esetén az adatvédelmi jog teljes szankciórendszere felhívható.

Meg kell jegyezni, hogy a közigazgatás egyes területén a jogalkotó az informatikai biztonság szabályozásában jelentősen továbbmegy az Infotv. szabályain. Az állami és önkormányzati szervek elektronikus információbiztonságáról szóló 2013. évi L. törvény az informatikai biztonság területén részletes szabályrendszert és felügyeleti szervezetrendszert állapít meg, amelyek természetesen nagyban hozzájárulnak az adatbiztonság szintjének növeléséhez.

## 6. Az érintett jogai az adatkezeléssel kapcsolatban

Az Infotv. személyes adatainak kezelésével kapcsolatban sajátos jogokat biztosít az érintettek részére, amelyek az adatkezelés egész folyamatában biztosítják az információs önrendelkezési jog érvényesítésének lehetőségét. Az érintett jogait kizárólag törvény korlátozhatja, az adatvédelmi törvényben meghatározott közérdekű célokból.[99]

- Az érintettet nemcsak az adatfelvételkor kell tájékoztatni az adatkezelés lényeges körülményeiről, hanem azokról az adatkezelés során is tájékoztatást kérhet. Az adatkezelő köteles a kérelem benyújtásától számított

98 Az Európai Gazdasági Térség az Európai Unió tagállamain kívül magába foglalja Izlandot, Lichtensteint és Norvégiát.
99 Gálik – Polyák im. 223.

a lehető legrövidebb idő alatt, legfeljebb azonban 30 napon belül írásban, közérthető formában megadni a kért tájékoztatást.
- Az érintett kérheti a valóságnak meg nem felelő személyes adatai helyesbítését. Ha a hiányos vagy téves adat jogszerűen nem korrigálható, akkor azt – törvény eltérő rendelkezése hiányában – törölni kell.
- A kötelező (törvényen alapuló) adatkezelések kivételével az érintett bármikor kérheti személyes adataink törlését, és ezzel az adatkezelés megszüntetését. Az adattörlés az adatok felismerhetetlenné tételét jelenti oly módon, hogy a helyreállításuk többé nem lehetséges. A törlésre irányuló kérelemnek nem feltétele, hogy az adatkezelés jogellenes legyen.[100]
- Az adatvédelmi törvény az érintett részére biztosítja továbbá a tiltakozás jogát, amelynek elsősorban az érdekmérlegelés alapján történő adatkezeléseknél van jelentősége. A tiltakozás az érintett olyan nyilatkozata, amellyel személyes adatainak kezelését kifogásolja, és az adatkezelés megszüntetését, illetve a kezelt adatok törlését kéri. E jogával az érintett különösen akkor élhet, ha – a kötelező adatkezelés kivételével – a személyes adatok kezelése kizárólag az adatkezelő jogának vagy jogos érdekének érvényesítéséhez szükséges, valamint ha a személyes adat felhasználása vagy továbbítása közvetlen üzletszerzés, közvélemény-kutatás vagy tudományos kutatás céljára történik. Az adatkezelő – az adatkezelés egyidejű felfüggesztésével – a tiltakozást köteles a kérelmet legfeljebb 15 nap alatt megvizsgálni, és annak eredményéről a kérelmezőt írásban tájékoztatni. Amennyiben a tiltakozás indokolt, az adatkezelő köteles az adatkezelést megszüntetni és az adatokat zárolni.[101]

Amennyiben az adatkezelő az érintettet korlátozza jogainak gyakorlásában, akkor az érintett a jogait bírósági úton is érvényesítheti. A bírósági eljárásra vonatkozó rendelkezések az érintett részére kedvező helyzetet teremtenek, többek között a bizonyítási kötelezettség megállapításával. A bizonyítási eljárás általános szabályaival szemben nem az érintett, hanem az adatkezelő köteles bizonyítani azt, hogy az adatkezelés a jogszabályban foglaltaknak megfelel. Ha a bíróság a kérelemnek helyt ad, az adatkezelőt a tájékoztatás megadására, az adat helyesbítésére, törlésére, az érintett tiltakozási jogának figyelembevételére kötelezi. A bíróság elrendelheti ítéletének nyilvánosságra hozatalát, ha azt az adatvédelem érdekei és nagyobb számú érintett jogai megkövetelik.[102]

## 7. Adatvédelmi szabályok a közigazgatásban

A közigazgatás rendszerén belüli adatkezelésekre, konkrétan a hatósági eljárásra vonatkozó adatvédelmi szabályokat a közigazgatási hatósági eljárás és szolgáltatás általános szabályairól szóló 2004. évi CXL. törvény (Ket.) tartalmazza. Ezeket egyes ágazatokra vonatkozóan további szektorális adatvédelmi szabályok egészítik ki.

A hatósági eljárás tartama alatt, illetve a hatóság által nyújtott szolgáltatás teljesítése során – különösen az iratokba való betekintés engedélyezésénél, a tárgyalás során, a döntés szerkesztésénél és a döntésnek hirdetményi úton való közlésénél – a hatóság gondoskodik arról, hogy a törvény által védett titok és a hivatás gyakorlásához kötött titok (a továbbiakban együtt: védett adat) ne kerüljön nyilvánosságra, ne juthasson illetéktelen személy tudomására, és a személyes adatok védelme biztosított legyen.

A hatóság az eljárása során, illetve az általa nyújtott szolgáltatás teljesítésekor jogosult a természetes személy ügyfél és az eljárás egyéb résztvevője azonosítása céljából
- nevének,
- születési nevének,
- születési idejének,
- születési helyének,
- anyja nevének,
- lakcímének, valamint
- külön törvényben vagy - törvény felhatalmazása alapján, az abban meghatározott körben - helyi önkormányzat rendeletében meghatározott személyes adatának kezelésére.

A hatóság törvény eltérő rendelkezése hiányában jogosult az eljárás lefolytatásához, illetve a szolgáltatás nyújtásához elengedhetetlenül szükséges személyes adatok megismerésére és kezelésére. A hatóság az eljárása, illetve szolgáltatása nyújtása során – jogszabályban meghatározott módon és körben – jogosult az eljárás lefolytatásához szükséges védett adat megismerésére.

A hatóság az eljárás, illetve szolgáltatás nyújtása során a személyes adatot és a védett adatot – az ugyanazon ügyben folyó, e törvényben szabályozott eljárások kivételével – csak akkor továbbíthatja más szervhez, ha azt törvény meg-

100 Infotv. 14-17. §§
101 Infotv. 21. §
102 Infotv. 22. §

engedi, vagy ha az érintett ehhez hozzájárult. A hozzájárulást az érintett az ügyintézési rendelkezésben, továbbá – ha jogszabály lehetővé teszi -, az érintett elektronikus azonosítását követően, az információ megőrzését biztosító módon történő rögzítés mellett telefonon is megadhatja.

A hatóság eljárásának jogerős befejezését követően az eljárása során megismert és kezelt személyes adatokat zárolja. A hatóság a zárolt személyes adatokat az eljárás tárgyát képező ügy iratainak selejtezéséig vagy levéltári őrizetbe adásáig tárolja, azokat kizárólag a jogerős döntésének végrehajtása, a jogerős döntésében foglaltak ellenőrzése vagy a jogerős döntésével összefüggő jogorvoslat vagy döntés-felülvizsgálat céljából kezelheti, és kizárólag e személyes adatok kezelésére jogosult más szerv vagy személy részére továbbíthatja.

A természetes személy ügyfelekre vonatkozó, az ügyfelekkel vagy más hatóságokkal való kapcsolattartásra használt azonosító adatokat a hatóság más adatoktól elkülönítve, olyan módon kezeli, hogy a nyilvántartások összekapcsolását mindkét nyilvántartáshoz történő jogosulatlan hozzáférés esetén is műszaki megoldás akadályozza meg.

Ha a hatóság különleges adatot az ügyfél hozzájárulása alapján továbbít, az ügyfél a hozzájárulását az alábbi módokon is megadhatja:

- hozzájárulás adása az ügyintézési rendelkezésben,
- hozzájárulás az ügyfél ügyintézési rendelkezésében meghatározott módon hitelesített elektronikus nyilatkozatában,
- ha jogszabály lehetővé teszi, hozzájárulás adása telefonon az ügyfél elektronikus azonosítását követően, az információ megőrzését biztosító módon történő rögzítéssel.[103]

## 8. Az adatvédelem felügyeleti és szankciórendszere

Az adatvédelmi előírások megsértésére összetett szankciórendszert biztosít a jogrendszer. Az adatvédelmi követelmények megsértése esetére a jogalkotó mind reparatív, mind represszív jellegű szankciókat kilátásba helyez.

Mivel az adatvédelmet a Ptk. a személyhez fűződő jogok közé sorolja, ezért azzal kapcsolatban az általános polgári jogi szankciók is igénybe vehetők. Az adatvédelmi törvény ugyanakkor az általánostól eltérő kárfelelősségi mércét tartalmaz.

Az adatvédelem büntetőjogi következményeit olyan ún. keretténállás szabályozza, amelyek a magatartás jogellenes jellegét az általános és az adott adatkezelésre vonatkozó ágazati adatvédelmi rendelkezések alapján rendelik meghatározni.[104]

Az adatvédelem területét emellett széles jogkörökkel rendelkező felügyelő hatóság, a Nemzeti Adatvédelmi és Információszabadság Hatóság (NAIH) is felügyeli.

### 8.1 Kártérítési felelősség és sérelemdíj

Az Infotv. az adatvédelmi követelmények megszegésével okozott károkért viselt felelősséget az általános polgári jogi felelősségnél szigorúbban szabályozza. A törvény szerint az adatkezelő az érintett adatainak jogellenes kezelésével vagy a technikai adatvédelem követelményeinek megszegésével másnak okozott kárt köteles megtéríteni. Ha pedig az adatkezelő az érintett adatainak jogellenes kezelésével vagy az adatbiztonság követelményeinek megszegésével az érintett személyiségi jogát megsérti, az érintett az adatkezelőtől sérelemdíjat követelhet. A kártérítési felelősség és a sérelemdíj megfizetésének kötelezettsége alól kizárólag akkor mentesül, ha bizonyítja, hogy a kárt az adatkezelés körén kívül eső elháríthatatlan ok idézte elő.[105] Nem kell megtéríteni a kárt annyiban, amennyiben az a károsult szándékos vagy súlyosan gondatlan magatartásából származott.[106]

### 8.2 Büntetőjogi felelősség

A Btk. Személyes adattal visszaélés című tényállása (Btk. 219. §) egyes adatvédelmi követelmények súlyosabb megszegésével szemben büntetőjogi szankciók alkalmazását rendeli. A bűncselekményt az követi el, aki a személyes adatok védelméről vagy kezeléséről szóló törvényi rendelkezések megszegésével haszonszerzési célból vagy jelentős érdeksérelmet okozva:

- jogosulatlanul vagy a céltól eltérően személyes adatot kezel,
- az adatok biztonságát szolgáló intézkedést elmulasztja, illetve

103 Ket. 17-17/A. §§
104 Gálik – Polyák im. 224.
105 Ez a szabályozás az adatkezelő felelősségét a polgári jog ún. veszélyes üzemi kárfelelősségi mércéje szerint szabályozza
106 Infotv. 23. §

aki az érintett tájékoztatására vonatkozó kötelezettségének nem tesz eleget, és ezzel más vagy mások érdekeit jelentősen sérti (a haszonszerzési céltól függetlenül).

Súlyosabban büntetendő a különleges személyes adatokkal való visszaélés, valamint hivatalos személyként, közmegbízatás felhasználásával elkövetett bűncselekmény.

## 8.3 A Nemzeti Adatvédelmi és Információszabadság Hatóság

Az új adatvédelmi szabályozás kétségkívül legtöbb vitát kiváltó eleme az adatvédelem és információszabadság felügyeleti rendszerének újraszabályozása. A szabályozás lényege, hogy – megtartva több adatvédelmi biztosi jogosítványt – új hatósági jogkörökkel és bírságolási joggal kiegészülve felügyeleti hatóság jött létre, amelynek elnökét a miniszterelnök javaslatára a köztársasági elnök nevezi ki. Az új hatóság 2012. január 1-i felállítása miatt az adatvédelmi biztos intézménye a hivatalban lévő adatvédelmi biztos mandátumának mintegy félidejénél megszűnt.[107]

A hatóság autonóm államigazgatási szerv, amelynek elnökét a miniszterelnök javaslatára a köztársasági elnök nevezi. A kinevezés módja a jelenlegi modellhez képest egyértelmű visszalépés: az országgyűlés kétharmadának szavazatával megválasztott adatvédelmi biztoshoz képest a hatóság elnöke kevésbé lehet független tisztség. A szervezetrendszerrel kapcsolatban több rendelkezés is biztosítja a formális függetlenséget: az elnök megbízása meglehetősen hosszú időre, kilenc évre szól, és a visszahívás feltételei korlátozottak és pontosan körülhatároltak. Az újraválaszthatóság lehetősége ugyanakkor nagymértékben rontja a hosszú kinevezési idő függetlenséget erősítő hatását. A törvény az elnökjelölttel szemben részletes összeférhetetlenségi szabályokat és szakmai követelményeket határoz meg (amelyek nem különböznek a korábbi, adatvédelmi biztosra vonatkozó szabályoktól).

A hatóság elnökének munkáját az általa határozatlan időre kinevezett elnökhelyettes segíti, amely felett az elnök gyakorolja a munkáltatói jogokat. Az elnökhelyettesre vonatkozó kinevezési feltételek és összeférhetetlenségi szabályok megegyeznek az elnökre vonatkozó szabályokkal. Mind az elnököt, mind a helyettesét vagyonnyilatkozat-tételi kötelezettség terheli.[108] A hatóság elnöke – meghatározott végzettséggel és gyakorlattal rendelkező – vizsgálókat nevezhet ki a köztisztviselői létszám legfeljebb húsz százalékáig.[109] A vizsgáló más köztisztviselőhöz viszonyított speciális jogállását, feladatait sem a törvényszöveg, sem az indokolás nem részletezi.

A törvény részletesen meghatározza a hatóság feladat- és hatásköreit és egyes hatásköröknél részletezi az eljárási szabályokat.

Az Infotv. 52. §-a szerint a hatóságnál bejelentéssel bárki vizsgálatot kezdeményezhet arra hivatkozással, hogy személyes adatok kezelésével, illetve a közérdekű adatok (közérdekből nyilvános adatok) megismeréséhez fűződő jogok gyakorlásával kapcsolatban jogsérelem következett be, vagy annak közvetlen veszélye fennáll. A hatóság – néhány kivételtől eltekintve – köteles a vizsgálatot lefolytatni,[110] az eljárás során a jelenlegivel lényegében azonos vizsgálati jogkörök illetik meg.[111]

A vizsgálat szabályozásával – az indokolás szerint is[112] – lényegében a korábbi ombudsman jellegű hatásköröket kívánta a jogalkotó átmenteni, és külön is rögzíti, hogy a hatóság vizsgálata nem minősül közigazgatási hatósági eljárásnak, arra a Ket. szabályait nem kell alkalmazni.[113]

A vizsgálat végeztével – 2 hónapon belül – a hatóság felhívhatja az adatkezelőt a jogsérelem orvoslására, amelynek megtételéről, vagy arról, hogy az abban foglaltakkal nem ért egyet, az adatkezelő 30 napon belül köteles a hatóságot tájékoztatni. Amennyiben ez nem vezet eredményre, úgy a hatóság ajánlást tehet a szerv felügyeleti szervének,[114] nyilvános jelentést készíthet az ügyről, illetve adatvédelmi hatósági eljárást kezdeményezhet vagy titokfelügyeleti hatósági eljárást kezdeményezhet.[115] E két hatósági eljárást egyébként a hatóság felhívás és/vagy ajánlás kibocsátása nélkül is kezdeményezheti.

A hatóság ugyancsak a vizsgálat eredményeként ajánlást tehet jogszabály módosítására is, amennyiben a jogsérelem vagy annak közvetlen veszélye a jogi szabályozás hiányosságára vagy fölösleges, nem egyértelmű rendelkezésére vezethető vissza.[116]

107 Magyarország ezzel a jogalkotói megoldással az Európai Bíróság ítélete alapján megsértette az Európai Unió adatvédelmi irányelvének tagállami hatóságok függetlenségére vonatkozó szabályait.
108 Infotv. 46-47. §§
109 Infotv. 51. § (1) bekezdése
110 A vizsgálat lehetséges elutasításának oka lehet továbbra is pl. az, ha az ügyben bírósági eljárás van folyamatban. A további szabályokat ld. 53. § (2)-(3) bekezdése
111 Infotv. 54. §
112 Infotv. indokolás, az 52. §-hoz
113 Infotv. 52. § (2) bekezdése
114 A felügyeleti szerv számára a jogsérelemre való közvetlen felhívás nélkül is tehet ajánlást.
115 Infotv. 56, 58. §§
116 Infotv. 57. §

Akár a vizsgálat eredményeként, akár vizsgálati eljárás nélkül is a hatóság adatvédelmi hatósági eljárást indít, ha valószínűsíthető a személyes adatok jogellenes kezelése, és a jogellenes adatkezelés

- személyek széles körét érinti,
- különleges adatokat érint, vagy
- nagy érdeksérelmet vagy kárveszélyt idézhet elő.[117]

A hatóság e feltételek fennállása nélkül is indíthat hatósági eljárást, ha a személyes adatok védelméhez fűződő jog érvényesítése érdekében.[118]

A hatósági eljárásra a Ket. szabályait az Infotv.-ben meghatározott eltérésekkel kell alkalmazni; az ügyintézési határidő két hónap. Az eljárás kizárólag hivatalból indítható, az akkor sem minősül kérelemre indult eljárásnak, ha bejelentésen alapuló vizsgálat előzte meg.[119]

Az adatvédelmi hatósági eljárásban szankciórendszere részben átveszi a jelenlegi – nagyrészt az EU adatvédelmi irányelvén alapuló – jogkövetkezményeket, kiegészítve a bírságolás lehetőségével. A hatóság határozatban

- elrendelheti a valóságnak nem megfelelő személyes adat helyesbítését,
- elrendelheti a jogellenesen kezelt személyes adatok zárolását, törlését vagy megsemmisítését,
- megtilthatja a személyes adatok jogellenes kezelését vagy feldolgozását,
- megtilthatja a személyes adatok külföldre továbbítását,
- elrendelheti az érintett tájékoztatását, ha azt az adatkezelő jogellenesen tagadta meg,
- 100.000 Ft-tól 10.000.000 Ft-ig terjedő bírságot szabhat ki,[120]
- valamint a hatóság elrendelheti a határozat – az adatkezelő azonosító adatainak közzétételével történő – nyilvánosságra hozatalát, ha azt az adatvédelem érdekeinek, illetve nagyobb számú érintett jogainak védelme ezt megköveteli.[121]

A bírság összegével kapcsolatban megjegyezzük, hogy a kiszabható legnagyobb összeg európai összehasonlításban és egyes adatkezelőkkel: például multinacionális cégekkel szembeni visszatartó erőt figyelembe véve is meglehetősen alacsony. Ennek indokáról az törvény indokolása hallgat; vélhetően a bírságolási lehetőség újdonságára tekintettel állapított meg a jogalkotó viszonylag alacsony összegeket.[122]

A hatóság másik nevesített hatósági jellegű, Ket. szabályai szerint lefolytatott eljárása a titokfelügyeleti hatósági eljárás, amely akkor indítható, ha a bejelentésen alapuló vizsgálat alapján vagy egyébként valószínűsíthető, hogy a nemzeti minősített adat minősítése jogellenes. Ezen eljárást szintén minden esetben hivatalból indítja meg a hatóság, akkor is, ha az eljárást vizsgálat előzte meg.[123]

Az eljárás eredményeként a hatóság a minősítésére vonatkozó jogszabályok megsértésének megállapítása esetén a minősítőt a minősítés szintjének, illetve érvényességi idejének a megváltoztatására, vagy a minősítés megszüntetésére hívhatja fel. A minősítő a határozat bírósági felülvizsgálatát hatvan napon belül kérheti. A bíróság eljárására a közigazgatási perekre vonatkozó rendelkezéseke kell alkalmazni azzal, hogy a bíróság az ügyben zárt tárgyaláson, soron kívül jár el.[124]

A fent részletezett eljárások mellett a hatóság

- javaslatot tehet az adatvédelmet és információszabadságot érintő jogszabályok megalkotására, illetve módosítására, valamint véleményezi a feladatkörét érintő jogszabályok tervezetét;
- általános jelleggel vagy meghatározott adatkezelő részére ajánlást bocsát ki;
- véleményezi a közfeladatot ellátó szerv tevékenységével kapcsolatosan az e törvény szerint közzéteendő adatokra vonatkozó különös, illetve egyedi közzétételi listákat;
- megszervezi a belső adatvédelmi felelősök konferenciáját;
- adatvédelmi nyilvántartást vezet;
- meghatározza az adatvédelmi auditálás szakmai szempontjait;
- az adatkezelő kérelmére adatvédelmi auditot folytathat le;
- tevékenységéről minden év március 31-ig beszámol az Országgyűlésnek.[125]

117 Infotv. 60. § (4) bekezdése
118 Infotv. 60. § (1) bekezdése
119 Ekkor azonban a bejelentőt az adatvédelmi hatósági eljárás megindításáról, illetve befejezéséről értesíteni kell. Ld. Infotv. 60. § (1)-(3), (5) bekezdése
120 A hatóság a bírság kiszabása során figyelembe veszi a jogsértéssel érintettek körének nagyságát, a jogsértés súlyát és a jogsértés ismétlődő jellegét. Infotv. 61. § (4) bekezdése
121 Infotv. 61. § (1)-(3) bekezdése
122 Polyák – Szőke im. 168-169.
123 Infotv. 62. §
124 Infotv. 63. § (1)-(3)
125 Infotv. 38. § (3)-(4)

Nemzeti
Közszolgálati Egyetem
Vezető- és Továbbképzési Intézet

BALOGH ZSOLT GYÖRGY

# Elektronikus dokumentumok kezelése, hitelesítése, megőrzése

Budapest, 2014

A tananyag az ÁROP – 2.2.21 Tudásalapú közszolgálati előmenetel című projekt keretében készült el.

**Szerző:**


**Kiadja:**


**Felelős kiadó:**
Patyi András
rektor





# 1. Az elektronikus aláírás

## 1.1 Az elektronikus aláírás technikai háttere

### 1.1.1 Bevezetés

Napjaink információs forradalma és a nyomában kiépülő információs társadalom alapvető technológiája az automatizált adatfeldolgozás és az elektronikus adatátvitel. E technológia hallatlan mértékben megnöveli az emberek közötti távolsági kommunikáció hatókörét és kiterjeszti alkalmazásának területét. Mind több írásbeli és verbális üzenetváltás zajlik a világban igen nagy földrajzi távolságban lévő helyek között, s tartalmukat tekintve ezek az üzenetek is nagyon sokfélék. Természetesen nagy súlyt képviselnek közöttük az olyan üzenetek, melyeket egymást jól ismerő felek váltanak, de emellett egyre növekszik az egymással baráti, rokoni vagy egyéb személyes kapcsolatban nem álló felek közötti üzleti vagy igazgatási célú üzenetváltások mennyisége is. Mivel a távközlési eszközök és rendszerek használata egyre nagyobb mértékben beépül a magánélet, a gazdasági kapcsolatok és az igazgatási ügyek intézésének gyakorlatába, természetesen egyre nagyobb az igény az iránt, hogy ez a kommunikáció biztonságos, esetenként akár hiteles és illetéktelen beavatkozástól mindenképpen mentes legyen.

Természetesen az írásbeliség több ezer éves története során az emberiség folyamatosan törekedett arra, hogy a rendelkezésére álló technológia mindenkori szintjén megvalósítsa a távollevő felek közötti hiteles és biztonságos kommunikációt. Az erre szolgáló megoldások jól ismertek. viaszpecsét, bélyegző, saját kezű aláírás illetve ennek tanúkkal történő igazolása, közjegyzői hitelesítés, rejtjelezés és még számos más eszköz.

Korunkban azonban nyilvánvalóan új technológia és – ne habozzunk kimondani – gyökeresen új típusú írásbeliség van kialakulóban, amely természetesen igényli a bizalom megteremtésének és fenntartásának új megoldásait is. Ennek a változásnak a hordereje talán csak az írás és hagyományos írásbeliség több ezer vagy a könyvnyomtatás ötszáz évvel ezelőtti feltalálásával mérhető össze. Az előző technikák esetében a hitelesítési megoldások létrehozására és finomítására kellően hosszú idő állt rendelkezésre. A hiteles elektronikus dokumentumok és hiteles elektronikus kommunikáció eszközeinek és eljárásainak kidolgozása útján azonban még csak az első lépéseket tettük meg. A továbbiakban ennek a folyamatnak az eddig elért eredményeiről lesz szó, különös tekintettel a joghatás kiváltására irányuló dokumentumok hitelesítésének problémájára.

### 1.1.2 Hiteles okiratokkal kapcsolatos követelmények

A jogrendszer – különösen a magánjog – egyik alapvető rendeltetése az *ügyleti forgalom biztonságának* garantálása. Ez gyakorlatilag elképzelhetetlen megbízható eljárásjogi intézmények, többek között a gyakorlati igényeket jól szolgáló bizonyítási rendszer nélkül. A kereskedelmi tranzakciók egyik lényeges alaki jellegzetessége az írásbeliség, ami a szerződések írásba foglalásától a számviteli műveletek bizonylatokkal történő dokumentálásáig terjed. Ezek a jogi tényeket tartalmazó okiratok valamint a pénzügyi bizonylatok a hagyományos kereskedelmi tevékenység elterjedése során alakultak ki és váltak általánosan alkalmazott bizonyítási eszközökké.

Ideális esetben az okiratok felhasználása az egyéb eszközökön alapuló, gyakran kétes kimenetelű, lassú, nehézkes és költséges bizonyítás lefolytatását – amely tanúk meghallgatásával, szemletárgyak megtekintésével, helyszíni vizsgálatokkal, esetleg szakértői vélemény beszerzésével jár – elkerülhetővé teszi. A gyakorlat a hagyományos okiratok vonatkozásában is olyan kezelési módszer kialakítására törekedett és törekszik ma is, amelynek keretében az irat valamely tény fennállását – vagy éppen fenn nem állását – gyorsan és egyszerűen képes igazolni. E rendeltetésüknek az okiratok akkor tudnak megfelelni, ha kezelésük és felhasználásuk során két feltétel teljesül:

*a.) Technikai biztonság*: A technológia és az írásbeliség adott fejlettségi szintjét tekintve az okirat maradandó eszközzel és kellően biztonságos technológiával készült. Olyan eszközök felhasználásával, melyek a gyakorlat számára elfogadható áron és megfelelő valószínűséggel biztosítják az illetéktelen beavatkozás kizárását vagy legalább azok felismerhetőségét. A technikai biztonság mindig arányossági kérdés is. Mérlegelni kell a biztonság megteremtésének költségeit és az elérni kívánt biztonsági szintet. Valójában sohasem beszélhetünk teljes biztonságról. csak a költségek és elvárások arányában definiálható *optimális biztonság* létezik.

*b.) Valódiság vélelme*: A fenti, kellően erős technikai védelemmel ellátott dokumentum tartalmát úgy kell tekinteni, hogy az helyesen és pontosan fejezi ki a ténybeli valóságot. Ez a vélelem mindaddig fennáll, amíg törvényes eljárás keretében az ellenkezője be nem bizonyosodik.

A hagyományos okiratok körében az aláírás vált mára a hitelesítés elsődleges eszközévé, amit a többi alkalmazott eszköz csak kiegészít és megerősít. Az aláírás általánosan elfogadott céljai az alábbiak:

**a.) Azonosíthatóság**: az aláíró személyének és magának az okiratnak az egyediesítése.
**b.) Hitelesség**: annak igazolása, hogy az okirat valóban attól származik, akit kibocsátójaként a szöveg megnevez.
**c.) Sértetlenség**: az irat tartalmi integritásának bizonyítása. Védelmet jelent az aláírás elhelyezését követően eszközölt módosítások ellen.
**d.) Bizonyító erő:** Letagadhatatlanság, azaz nyilatkozattétel tényének rögzítése, megerősítése.

Az elektronikus kereskedelem és az elektronikus közigazgatás egyre szélesebb körben terjedő szolgáltatásainak köszönhetően gyorsan felmerült az igény a hiteles elektronikus okiratok alkalmazása iránt annak érdekében, hogy ebben a körben is teljesülhessen az ügyleti forgalom biztonságának követelménye, azaz lehetővé váljon a hiteles elektronikus nyilatkozattétel. Korábban az elektronikus okiratok esetében gyakorlatilag nem lehetett a fenti feltételeket kielégíteni. Speciális megoldások nélkül az elektronikus iratok világából minduntalan vissza kellett térnünk a hagyományos hitelesítés eszközeihez és megoldásaihoz. A gyakorlatban ez azt jelentette, hogy azokat az elektronikus úton létrehozott dokumentumokat, amelyek joghatás kiváltására irányultak ki kellett nyomtatni és ezután a hagyományos módszerekkel – aláírással, bélyegzővel – hitelesíteni és a továbbiakban is manuális módszerekkel feldolgozni. Ez nemcsak felesleges párhuzamosságot okoz a dokumentumkezelésben, hanem a papír nélküli iroda és az ettől remélt gyorsabb és hatékonyabb ügyintézés kialakítását is kizárja.

### 1.1.3 Minek nevezzelek? Elektronikus aláírás és/vagy digitális aláírás

*Elektronikus aláírásnak* tekinthetünk minden olyan megoldást, aminek segítségével a kibocsátó elhelyezi az aláírását egy elektronikus dokumentumon. Tágabb értelemben így természetesen elektronikus aláírás az is, ha a nevét, megnevezését egyszerűen odagépeli a dokumentum végére, vagy a hagyományos papíralapú aláírásról készült digitalizált képet beilleszti a dokumentumba. Ezeknek a megoldásoknak a bizonyító ereje azonban nem tesz eleget a jogrendszer által támasztott hitelességi követelményeknek.

Az elektronikus dokumentumok esetében ma rendelkezésünkre álló technológia valójában a *digitális aláírás*, amely joghatás kiváltására irányuló dokumentumok hitelesítésére alkalmas. A továbbiakban erről a szűkebb értelemben vett elektronikus aláírási technológiáról, és a hozzá kapcsolódó szabályozási környezetről lesz szó. A szakszerűség és a pontosság érdekében helyes volna a „digitális aláírás" terminológia használata, mivel azonban a jelenleg hatályos törvény az „elektronikus aláírásról" szól, ez a tananyag az egységesség érdekében a törvény szóhasználatát követi, azzal a megjegyzéssel, hogy valójában a digitális aláírásról, tehát a szűkebb értelemben vett elektronikus aláírásról lesz szó.

Az ilyen elektronikus aláírás lehetővé teszi, hogy az elektronikus iratok teljes kezelési ciklusuk során megmaradjanak elektronikus formában. Az elektronikus aláírás a kapcsolódó eljárásokkal együtt alkalmas arra, hogy biztosítsa az aláíró egyértelmű azonosíthatóságát, az aláírás tényének letagadhatatlanságát, továbbá azt, hogy az elektronikus úton aláírt elektronikus irat tartalma nem változott meg azóta, hogy az a személy, akihez az elektronikus aláírás tartozik, az aláírást „elhelyezte" az iraton. Azon technológia biztonságosságának, amellyel az elektronikus aláírást előállítják, és hozzáfűzik az irathoz, jogi relevanciája is van. Ennek ellenére a szabályozás legfelső szintjének az alkalmazott technológiától függetlennek kell lennie, mivel az alkalmazott technológiák gyorsan változnak. A jogi szabályozás kidolgozásakor azonban mégsem lehet teljesen figyelmen kívül hagyni a jelenleg elfogadott technikai megoldásokból eredő követelményeket. Az elvileg rendelkezésre álló műszaki illetve matematikai megoldások közül az ún. *nyilvános kulcsú eljárásokra* alapozott elektronikus aláírás terjedt el.

### 1.1.4 Rejtjelezés és elektronikus aláírás

A nyilvános kulcsú hitelesítési eljárások a kriptográfián, azaz a rejtett tartalmú üzenetváltások technológiáján alapulnak. A rejtjelezés célja az, hogy a bizalmas üzenetet úgy juttassuk célba, hogy annak tartalmához csak a címzett

férhessen hozzá, és a *címzett* meggyőződhessen a *feladó* kilétéről. Ugyanakkor a rejtjelezés során fel kell tételezni azt is, hogy a címzetten, és a feladón kívül létezik egy harmadik személy, a *lehallgató*, aki

a.) megpróbál illetéktelenül hozzáférni az üzenet tartalmához, és/vagy
b.) a feladó nevében hamis üzenetet próbál küldeni a címzettnek.

A rejtjelezést a diplomáciai, a védelmi, katonai, a bizalmas természetű üzleti és a magán célú kommunikációban egyaránt régóta – valójában több ezer éve – alkalmazzák. E hosszú idő alatt a kriptográfiai algoritmusoknak két nagy osztálya alakult ki, a *szimmetrikus és az aszimmetrikus eljárások*.

#### 1.1.4.1 Szimmetrikus rejtjelezés

A szimmetrikus eljárások lényegi elemét képezi az a körülmény, hogy a feladó és a címzett ismernek valamilyen titkos információt, mindketten ugyanazt, amelyet a lehallgató nem ismer. Ennek a titkos információnak a birtokában képes a feladó a *rejtjelezett üzenetet létrehozni a nyílt szövegből*, a címzett pedig dekódolni tudja azt. A lehallgató, a titkos információ hiányában sem megfejteni nem tudja a rejtjelezett szöveget, sem arra nem képes, hogy maga hozzon létre és küldjön hamis tartalmú rejtjelezett üzenetet. Ennek alapján a címzett biztos lehet benne, hogy az üzenet valóban az általa ismert feladótól származik. A titkos információ általában a rejtjelezéshez használt algoritmus egyik paramétere, a *kódoló/dekódoló kulcs*.

Bár a szimmetrikus kulcsú rejtjelezésre hosszú évszázadok során nagyon jó módszereket dolgoztak ki, vannak kétségtelen hátrányai is. Az első számú gyakorlati problémát az jelenti, hogy a titkos kulcsban meg kell állapodnia a címzettnek és a feladónak. Ez egy olyan kommunikációs művelet, amelyet kellően biztonságos módon kell lebonyolítani, s ennek a megszervezése gyakran komoly nehézségeket támaszt. A titkos kulcsot olyan módon kell a feladónak és a címzettnek egymással megosztani, hogy eközben a lehallgató ne tudja azt megszerezni. Ehhez olyan kommunikációs csatornát kell használniuk, amely biztonságos, nem lehallgatható. Ennek a csatornának a használata azonban valamilyen értelemben korlátozott – nagyon drága vagy nagyon lassú – ellenkező esetben ugyanis nem kellene rejtjelezést használni, hanem ezen a csatornán lehetne az üzenetet küldeni.

Ugyancsak a kulcskezeléssel kapcsolatos probléma, hogy a rejtjelezett üzenetváltásban résztvevő minden kommunikáló párnak – minden egyes feladónak minden egyes címzettel – külön-külön meg kell egyeznie egy, vagy több titkos kulcsban. Ez már viszonylag kis számú pár esetén is sok bizalmasan megőrzendő kulcsot, következésképpen komoly szervezési és adminisztrációs feladatot jelent. Szabatosan kifejtve ugyanis $n$ partner esetén, ha mindenki mindenkivel kommunikálni fog, akkor $n(n-1)/2$ kulcsban kell megegyezni. Azaz ha $n=10$, akkor a szükséges kulcsok száma $10*9/2=45$.

További igen lényeges problémája a szimmetrikus kulcsú rejtjelezési algoritmusoknak, hogy bármelyik ismert és gyakorlatban alkalmazott megoldásnál kellően hosszú, vagy elegendően nagy számú üzenet alapján a kulcs – elsősorban statisztikai alapon – megfejthető, s a továbbiakban ennek felhasználása már nem biztonságos. Emiatt a kulcsokat gyakran cserélni kell, ami tovább fokozza a kulcskezeléssel kapcsolatos adminisztrációs és szervezési nehézségeket és az ebben rejlő biztonsági kockázatokat is.

A fenti problémák az aszimmetrikus rejtjelezési módszerekkel lényegesen mérsékelhetők.

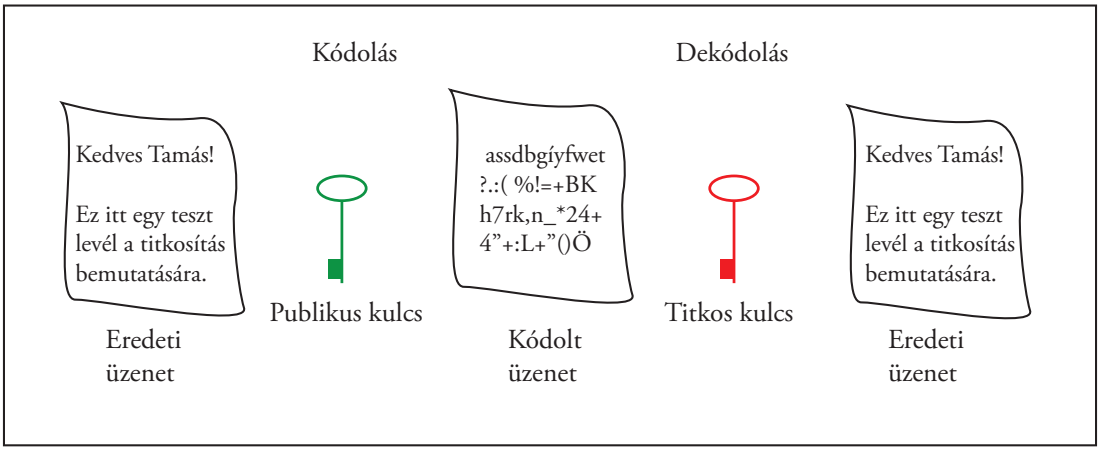
#### 1.1.4.2 Aszimmetrikus rejtjelezés

Az aszimmetrikus rejtjelezés története sokkal rövidebb, mint az előző módszeré. Az algoritmust elsőként James H. *Ellis*, Clifford *Cocks* és Malcolm *Williamson*, a *Government Communications Headquarters* (GCHQ) munkatársai dolgozták ki az 1970-es évek elején, eredményeiket azonban 1997-ig nem ismerhette meg a világ, s így az nem is hasznosulhatott. Az eljárás felfedezésének pillanatától államtitkot képezett, ugyanis a GCHQ a brit titkosszolgálatok egyike. A sors furcsa fintora, hogy ezúttal az elsőség nem garantálta a felfedezők számára a szakmai dicsőséget és a világhírt.

A titokban tartott eljárást újra fel kellett fedezni. Ez a néhány évvel későbbi eredmény *Diffie-Hellmann kulcscsere eljárásként* vált ismertté, miután 1976-ban két amerikai matematikus, Whitfield Diffie és Martin Hellman már civil felhasználásra szánt módszerként nyilvánosságra hozta. Ez az első gyakorlatban is alkalmazható megoldás olyan biz-

tonságos rejtjelkulcs megosztási rendszerre, amelyet nyilvános célú – tehát lehallgatás ellen nem védett – kommunikációs csatornán keresztül valósítanak meg.

Az eljárás a kódolás, vagyis a rejtjel létrehozásának műveletét elválasztja a dekódolástól, és olyan algoritmust használ, ahol a kódoláshoz használt kulcs nemcsak, hogy nem azonos a dekódoláshoz használt kulccsal, hanem a kulcsok nem is határozhatók meg egymásból. Egy ilyen algoritmussal a kódoló kulcsot nyilvánossá téve bárki rejtjelezett üzenetet tud küldeni a címzettnek, akihez az a nyilvános kulcs tartozik, de dekódolni csak a címzett tudja, amennyiben a privát kulcsot csak az ő maga ismeri.




*1. ábra Kétkulcsos kriptográfia alkalmazása rejtjelezett üzenetváltásra*

A kulcskezelés adminisztrációja is átalakul és egyszerűsödik egy ilyen rendszerben. Ezt a módszert használva ugyanis létrehozható egy olyan nyilvánosan hozzáférhető kódkönyv, amiben mindenkinek szerepel a nyilvános kulcsa, és így a kódkönyvben szereplők bármelyike tud bárki másnak levelet küldeni anélkül, hogy előtte a partnerrel saját titkos kulcsban megállapodtak volna. Természetesen a kódkönyvnek megbízhatónak kell lennie.

A Diffie-Hellmann eljárás csak a kulcskezelés logikai sémáját képezi. A kívánt tulajdonságokkal rendelkező kulcsokat előállító szabatos és megbízható matematikai eljárást csak két évvel később, 1978-ban publikálták. Az *RSA algoritmus*, amely az új eredményt bejelentő három amerikai matematikus, Ronald *Rivest*, Ron *Shamir* és Len *Adleman* nevének kezdőbetűit őrzi, valójában Clifford Cocks módszerének általánosítása.

### 1.1.5 Elektronikus aláírás PKI technológiával

A ma használt elektronikus aláírási rendszerek az aszimmetrikus rejtjelezési technológián alapulnak. A felhasználás célja azonban meghatározó a felhasználás módja tekintetében is. Az elektronikus aláírásnak az a feladata, hogy a dokumentumot hitelesítse, nem pedig az, hogy illetéktelenek számára hozzáférhetetlenné tegye. Az eltérő célok pedig azt eredményezik, hogy ugyanazt az apparátust más módon használjuk fel. A legfontosabb különbség az aláírási folyamat eredményén azonnal észlelhető is. Az elektronikus aláírással ellátott dokumentum továbbra is nyílt hozzáférésű marad, tartalmaz azonban egy olyan adategységet, amely biztosítja és tanúsítja a hitelességét, a sértetlenségét és a letagadhatatlanságát.

Az aláírási rendszerben is jelen van az aszimmetrikus rejtjelezést lehetővé tevő kulcspár, de a rendszernek számos további eleme is van, melyek a hitelességet szolgálják. A technológia megfelelő működését különböző eljárásrendek és szervezetek biztosítják, amelyeket együttesen *Nyilvános Kulcsú Infrastruktúrának* (*Public Key Infrastructure*. **PKI**) nevezünk.

A PKI főbb elemei:

a.) az aláíró és az aláírást ellenőrző felek,

b.) a rendszert működtető megbízható harmadik fél (Trusted Third Party, **TTP**), azaz a *hitelesítés-szolgáltató*,
c.) a már ismert kriptográfiai nyilvános/titkos *kulcspár*,
d.) a kulcspár és az aláíró összetartozását bizonyító *tanúsítvány*.

#### 1.1.5.1 Az aláíró és az aláírást ellenőrző felek

Az egymással elektronikus úton kommunikáló és dokumentumokat cserélő felek természetesen alapvető elemei az elektronikus aláírási rendszernek. Aláíró fél az, aki a kulcspárral rendelkezik, akinek a javára elektronikus tanúsítvány került kiállításra s kriptográfiai privát kulcsának felhasználásával készíti el az elektronikus úton létrehozott dokumentumhoz tartozó elektronikus aláírást. Az elektronikus aláírással ellátott dokumentum címzettje ellenőrzi az aláírás, ezen keresztül pedig a kapott dokumentum hitelességét, az aláíró személyazonosságát. Az ellenőrzés elvégzéséhez az aláíró nyilvános kulcsát használhatja fel

#### 1.1.5.2 A hitelesítés-szolgáltató

A *hitelesítés-szolgáltató* olyan, a tanúsítvány kibocsátásra specializálódott szervezet, amely arra vállalkozik, hogy egy adott földrajzi területen – tipikusan egy országban – működő aláírókat az adott terület jogi, közigazgatási, gazdasági rendszerének ismeretében felépített eljárásrend alkalmazásával hitelt érdemlően azonosítsa, és ezek alapján igazolja akár az egész világ felé létezésüket és adataik valódiságát.

A hitelesítés-szolgáltató köteles az aláírók nyilvános kulcsait nyilvánosan kezelni és kérelemre rendelkezésre bocsátani, az aláíró ügyfelekről és a kiadott kulcsokról megbízható, magas szintű technikai védelemmel ellátott nyilvántartást vezetni, s a kulcsot az aláíró kérésére, valamint bíróság, vagy más erre jogosult szerv határozata alapján érvényteleníteni, a visszavont, érvénytelenített kulcsokról ugyancsak nyilvántartást vezetni.

A rendszerben kiemelt jelentőségű a hitelesítés-szolgáltató magánkulcsának védelme, mivel ezzel az eszközzel fogja a szolgáltató az általa kibocsátott tanúsítványokat hitelesíteni, azaz elektronikusan aláírja azokat. A hitelesítés-szolgáltató létezését önmaga igazolja önmaga számára kibocsátott tanúsítvánnyal. Az ehhez az úgynevezett *főtanúsítványhoz* kapcsolódó bizalmat a szervezet – a pénzintézetekhez hasonlóan – hagyományos üzletpolitikai eszközökkel teremti meg. Ezek közül a legfontosabbak a következők:
a.) átlátható, stabil, erős pénzügyi hátterű működés,
b.) büntetlen előéletű, magas szakmai felkészültségű vezetők és munkatársak,
c.) minőségvédelmi auditok elvégeztetése, ezek eredményének nyilvánosságra hozatala,
d.) bekerülés a nemzetközi hitelesítés-szolgáltatói listára,
e.) felelősségbiztosítás,
f.) pénzügyi garancia nyújtása a tanúsítványokkal történő esetleges visszaélésből eredő kár megtérítésére.

#### 1.1.5.3 A kulcspár

Az elektronikus aláírásnál alkalmazott kriptográfiai magánkulcs nyilvános kulcs olyan digitális jelsorozatok, amelyek megfelelő számítógép-programokkal kezelhetőek. A magánkulcsot az aláírónak titokban kell tartania, megfelelő intézkedésekkel védenie kell az illetéktelen felhasználás, eltulajdonítás ellen. Ha a magánkulcs titkossága megszűnik, akkor az eljárás garanciái (aláírt irat hitelessége, az aláíró azonosítása) megszűnnek. A megfelelően gondos kezelés a kulcstulajdonos felelőssége éppúgy, mint a bankkártyák esetében. A kulcs sokféle fizikai adathordozón tárolható. Az aláíró számítógépén található rejtjelezett adatállományban, flash-memórián, önálló vagy mobil telefonba épített chip kártyán. A nyilvános kulcs bárki által hozzáférhető, sőt kívánatos, hogy minél szélesebb körben megismerhető legyen (pl. a hitelesítés-szolgáltatónál vagy címtár-szolgáltatónál), ugyanis az aláíró a nyilvános kulcshoz kapcsolt adatok alapján azonosítható.

A kulcspár létrehozásakor megbízható matematikai algoritmusok biztosítják azt, hogy a nyilvános kulcs ismeretében gyakorlatilag nem lehet visszafejteni a hozzá tartozó magánkulcsot, illetve az aláírást nem lehet a nyilvános kulcs ismeretében hamisítani. A nyilvános kulcs ismeretében egy aláírásról gyakorlatilag egyértelműen megállapítható, hogy a hozzá tartozó magánkulcs felhasználásával készült-e vagy sem.

#### 1.1.5.4 A tanúsítvány

A rendszer működésének egyik sarokköve, hogy megbízható harmadik fél vállaljon garanciát az egész külvilág felé arra, hogy adott kulcspár, adott aláíróhoz tartozik. Erre szolgál a *tanúsítvány*, egy olyan elektronikus dokumentum, amely tartalmazza az aláíró szokásos azonosító adatait – név, lakcím illetve székhely, különböző nyilvántartási azonosító számok – és az aláírás ellenőrzéséhez használható nyilvános kulcsát. A tanúsítványban szereplő azonosító adatokat a hitelesítés-szolgáltató – hagyományos okiratok alapján – ellenőrzi, majd az ennek alapján összeállított elektronikus dokumentumot saját elektronikus aláírásával hitelesíti.

A hitelesítés-szolgáltató működésének és a hiteles tanúsítványnak köszönhetően az elektronikus úton aláírt iratok cseréje során nincs szükség a küldő és címzett személyes találkozására. Az elektronikus tanúsítványt hitelesíti az aláíró személyét és az elektronikus úton aláírt üzenetet, iratot, a címzett pedig egyértelműen meggyőződhet azok hitelességéről.

### 1.1.6 Az elektronikus aláírás működése

A nyilvános kulcsú eljárásokra alapozott elektronikus aláírás fő tulajdonságai:

a.) egy adott elektronikus aláírásnak csak egy és kizárólag egy tulajdonosa lehet,
b.) lehetővé teszi az aláíró személyének egyértelmű meghatározását,
c.) az aláírónak kizárólagos lehetősége van aláírásának készítésére,
d.) egyértelműen megállapítható az elektronikus úton aláírt elektronikus irat bármely, az aláírást követő véletlen vagy szándékos módosulása,
e.) további technológiai elemek, illetve szolgáltatások alkalmazásával az aláírás hiteles időpontja is csatolható az elektronikus úton aláírt irathoz.

Az elektronikus aláírás működése két lépésben valósul meg. Az első az aláírás létrehozása, a második pedig az aláírás ellenőrzése.

#### 1.1.6.1 Az elektronikus aláírás létrehozása

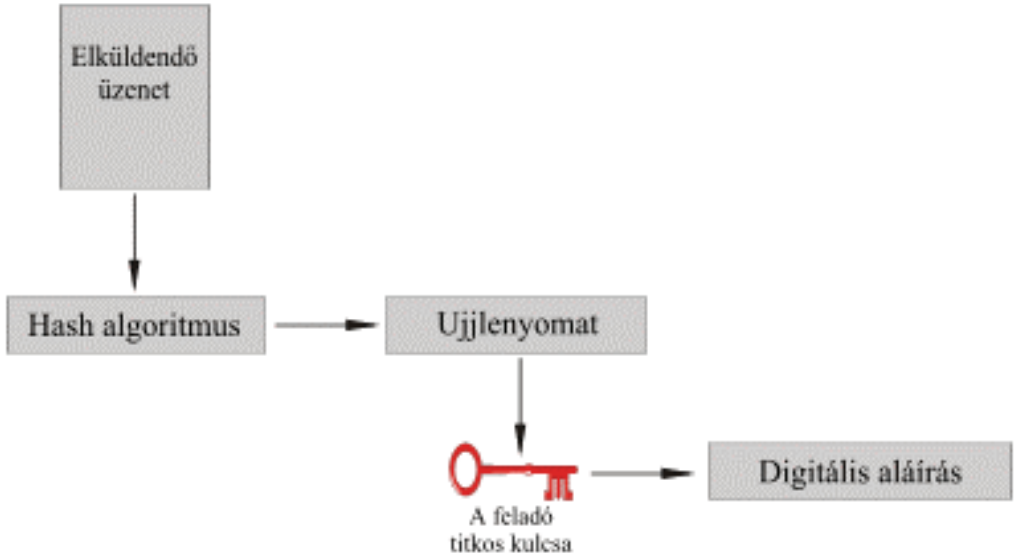
Az aláírás hitelesítő szerepe csak akkor valósulhat meg, ha a címzett kétséget kizáróan meggyőződhet arról, hogy az eredeti dokumentum jutott-e el hozzá, vagy annak egy módosult, manipulált változata. Problémát okoz, hogy külön védekezés nélkül a digitálisan tárolt adatoknál a változtatás nem hagy nyomot. A megoldást egy *digitális lenyomat* készítése és csatolása adhatja meg. A lenyomatkészítés az egyirányú hash-függvényen alapul.

##### 1.1.6.1.1 Lenyomat képzése hash-eljárással

A hash-függvény lényegileg egy olyan matematikai transzformáció, amely egy tetszőleges méretű – de véges – dokumentumból olyan fix hosszúságú bitsorozatot (digitális lenyomatot) állít elő, amely egyértelműen jellemző az adott szövegre. Eszerint tehát egy adott dokumentum lenyomata – valahányszor csak végrehajtjuk a transzformációt – mindig bitről bitre ugyanaz lesz. A lenyomat igen rövid, előre meghatározott hosszúságú bitsorozat. A jelenleg elterjedt szabványos eljárások – az alkalmazott algoritmustól függően – tipikusan 128 vagy 160 byte-os lenyomatokat eredményeznek. Ebből következik, hogy az egymástól különböző lenyomatot produkáló dokumentumok száma $256^{128}$ illetve $256^{160}$ lehet, ami a gyakorlat számára megnyugtató nagyságrend. E szerint ugyanis annak valószínűsége, hogy két dokumentum esetén a transzformáció azonos eredményt, azonos lenyomatot ad, 1: $256^{128}$ illetve 1:$256^{160}$.

A hash-transzformáció fő tulajdonságai az egyirányúság, az ütközésmentesség és az ún. lavinahatás:

a.) Egyirányúság: A lenyomatból az eredeti dokumentum nem állítható vissza és annak tartalmára sem lehet következtetni. Gyakorlatilag kizárt egy adott lenyomathoz olyan dokumentumot létrehozni illetve hozzárendelni, amelyik a transzformáció végrehajtásakor ugyanazt a lenyomatot produkálja.
b.) Ütközésmentesség: Gyakorlatilag lehetetlen két olyan dokumentumot konstruálni, amelyek azonos lenyomatot eredményeznek.
c.) Lavinahatás: Ha egy bitet megváltoztatunk a dokumentumban, akkor a lenyomat képe a bitek körülbelül felében különbözni fog.




*2. ábra Kétkulcsos kriptográfia alkalmazása elektronikus aláírás létrehozására*

#### 1.1.6.1.2 A lenyomat rejtjelezése

A dokumentumból képezett szabványos hosszúságú digitális lenyomatot a következő lépésben az aláíró saját kriptográfiai magánkulcsával kódolja. Az ennek eredményeként előállt rejtjelezett adathalmaz maga az elektronikus aláírás, amely az irathoz csatolva kerül továbbításra, vagy tárolásra. Ugyancsak tárolásra illetve továbbításra kerülhet – az aláíró tanúsítványának részeként – az aláírás ellenőrzésére szolgáló nyilvános kulcs is.

### 1.1.6.2 Az aláírás ellenőrzése

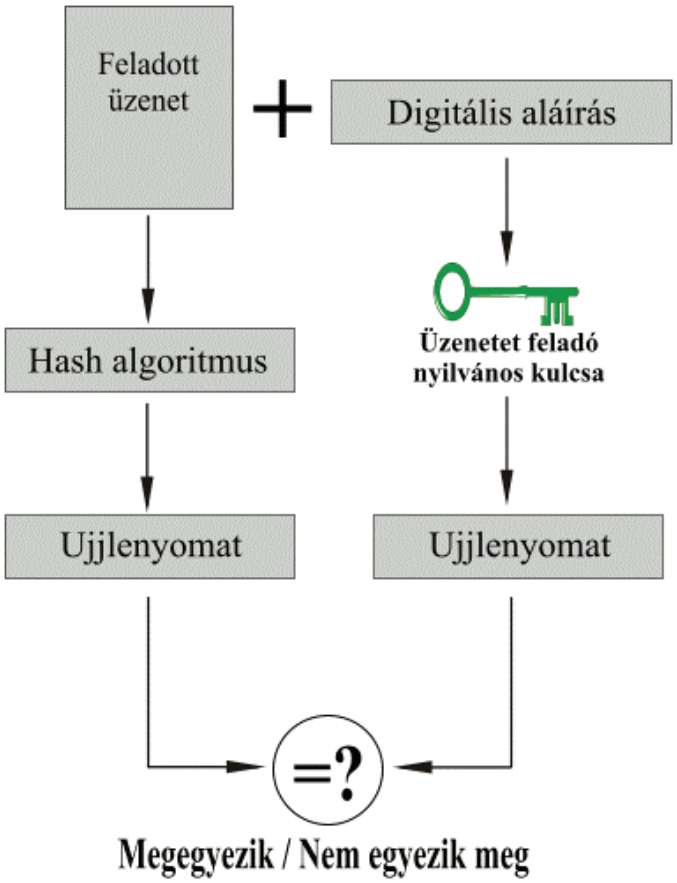
Az elektronikus aláírás ellenőrzését tipikusan az üzenet címzettje, a dokumentum jogosult felhasználója végzi. A címzett az ellenőrzés során – esetenként a hitelesítés-szolgáltató segítségével – meggyőződhet az elektronikus úton aláírt irat hitelességéről, és az aláíró személyéről. Az aláírás ellenőrzését, csakúgy, mint az aláírás létrehozását, e feladatra specializált számítógép-programok végzik el. Mind az aláíró mind a címzett csak kezdeményezi a műveletek elvégzését. A számítási műveleteket a szoftver végzi el, majd a folyamat végén közérthető, felhasználóbarát formában közli az eredményt.

Aláírás ellenőrzésének lépései:

a.) A program – a hash-függvény alkalmazásával – ismét létrehozza a dokumentum digitális lenyomatát.
b.) A szoftver a dokumentumhoz csatolt digitális aláírást a nyilvános kulcs felhasználásával dekódolja, amelyből szintén egy digitális lenyomat képződik.
c.) Az így előállított két digitális lenyomatot a rendszer bitről bitre összehasonlítja. Ha a lenyomatok azonosak, igen nagy valószínűséggel állítható, hogy az elektronikus úton aláírt irat tartalma nem változott továbbítása illetve tárolása során és az adott nyilvános kulcshoz tartozó magánkulccsal készült a digitális aláírás.

A fenti lépések mellőzhetetlenek az aláírás érvényességének megállapításához. A további két ellenőrzési lépésre csak akkor kerül sor, ha az aláíró személyazonosságát vagy a tanúsítvány érvényességét illetően kétség merül fel.

a.) Az aláíró személyét azonosító adatokat a nyilvános kulcs ismeretében a hitelesítés-szolgáltató kérelemre megadja és tanúsítja, és azt is igazolja, hogy az adott kulcs érvényes-e vagy sem (lejárt, ellopták stb.).
b.) Az aláíró tanúsítványának hitelessége szintén ellenőrizhető az azt kiállító hitelesítés-szolgáltató aláírásának a közismert, könnyen hozzáférhető nyilvános kulcsa alapján.




*3. ábra Kétkulcsos kriptográfia alkalmazása elektronikus aláírás ellenőrzésére*

### 1.1.7 Az elektronikus aláírás és a rejtjelezés felhasználási területei

Az elektronikus aláírásnak és a digitális rejtjelezési technikának számos alkalmazási területe van. Ezek köre egyre bővül, így e helyen csak néhány fontos és jellemző területet emelünk ki:

a.) Elektronikus levelezés biztonságossá tétele az üzenet aláírásával és rejtjelezésével
b.) Biztonságos internetes szolgáltatások, elsősorban elektronikus kereskedelmi illetve elektronikus közigazgatási szolgáltatások nyújtása a szolgáltató és az igénybe vevő megbízható azonosításával és a szolgáltatott adatok hitelességének, sérthetetlenségének, letagadhatatlanságának és bizalmasságának biztosításával.
c.) Zárt internet-közösségek létrehozása (orvosi konzílium, ügyvédi fórum, valamely előfizetett szolgáltatás) a felhasználók megbízható azonosításával és a kommunikáció bizalmassá tételével.
d.) Zárt kommunikációs csoportok kialakítása bármilyen kommunikációs felületen (mobiltelefon, Internet, LAN, WAN hálózatok) a csoport tagjainak azonosításával és az adatforgalom aláírásával, titkosításával.
e.) Biztonságos elektronikus fizetési módszerek kialakítása a felhasználó azonosításával és a készpénz-helyettesítő fizetési eszköz (tipikusan kártya) adatainak titkosításával akár Interneten akár mobiltelefonon keresztül.
f.) Elektronikus pénztárca és elektronikus pénz megoldások kis összegű vásárlások (micro payment) céljára a „pénz”-kibocsátó általi elektronikus aláírással.
g.) Biztonságos archiválás és adatbázis-kezelés, az eltárolt információ elektronikus aláírásával és/vagy titkosításával.
h.) Felhasználók biztonságos azonosítása és beléptetése informatikai eszközök általa védett rendszerekbe vagy helyiségekbe.
i.) Szoftverek származásának igazolása és sértetlenségének biztosítása a programkód aláírásával.
j.) Adatbázis vagy egyéb digitálisan tárolt tartalom – esetleg audiovizuális tartalom (fotó, hang- vagy videofelvétel) – létezésének, származásának igazolása és sértetlenségének biztosítása a fájl aláírásával.
k.) Különböző elektronikus okiratokkal kapcsolatos szolgáltatások.

## 1.2 Az elektronikus aláírás szabályozása és jogi alkalmazása

Az elektronikus aláírás szabályozási kereteit nemzetközi és hazai jogszabályok együttesen alkotják. Az Európai Unió 1999. december 13-án bocsátotta ki az elektronikus aláírásra vonatkozó közösségi keretfeltételekről szóló 1999/93/EK irányelvet. A tagállamok ennek alapján dolgozták ki saját belső nemzeti szabályozásukat.

Az irányelv állást foglal néhány lényeges kérdésben. Így biztosítja, hogy a kulcsfontosságú hitelesítés-szolgáltatást nyújthatja állami szerv, illetve természetes vagy jogi személy, amennyiben ez utóbbit a nemzeti jognak megfelelően hozták létre. Elvárja ugyanakkor, hogy a tagállamok gondoskodjanak olyan akkreditációs rendszer kialakításáról, amely a szolgáltatók minőségi és törvényességi kritériumoknak való megfelelés**ét felügyeli.**

**Az irányelv hozzájárul az elektronikus aláírások közösségen belüli alkalmazásához és jogi elismeréséhez.** Az elektronikus hitelesítési módszerek általános elfogadásának támogatása érdekében biztosítani kell, hogy az elektronikus aláírásokat minden tagállamban bizonyítékként lehessen felhasználni a bírósági eljárásokban, ugyanakkor az irányelv nem sérti a bizonyítékok szabad bírói mérlegelésére vonatkozó nemzeti szabályokat. Ezen túlmenően a nemzeti jog határozza meg, hogy a jog mely területein lehet elektronikus dokumentumokat és elektronikus aláírást alkalmazni. A hitelesítés-szolgáltatókra vonatkozó felelősségi szabályokat ugyancsak a nemzeti jog állapítja meg.

A hazai szabályozás gerincét az elektronikus aláírásról szóló 2001. évi XXXV. törvény [**Eat.**], a polgári perrendtartásról szóló 1952. évi III. törvény [**Pp**.] vonatkozó szakaszai, valamint a Polgári törvénykönyvnek [**Ptk**] a jognyilatkozatokról – különösen az írásbeli alakhoz kötött nyilatkozatok érvényességéről – szóló rendelkezései alkotják. A polgári jogot érintő közelmúltban lezajlott jelentős jogalkotási munkálatokra is tekintettel kell lenni, így figyelembe vesszük a régi Ptk [1959. évi IV. törvény] és az új Ptk [2013. évi V. törvény] felfogása közötti különbségeket is.

### 1.2.1 Alapfogalmak

Az Eat. több alapvető fogalom-meghatározást tartalmaz.

A törvény értelmében *elektronikus dokumentum*: elektronikus eszköz útján értelmezhető adategyüttes, azaz nemcsak írott dokumentum, hanem állókép, mozgókép, hangfelvétel, stb. is lehet elektronikus dokumentum (természetesen digitális formátum esetén), vagy akár szoftver is, és így ezek elektronikus aláírása is elképzelhető.

Az *elektronikus aláírás* a törvény szerint (az elektronikusan aláírt) elektronikus dokumentumhoz azonosítás céljából logikailag hozzárendelt vagy azzal elválaszthatatlanul összekapcsolt elektronikus adat.

Az *aláírás-létrehozó adat*: olyan egyedi adat (jellemzően kriptográfiai magánkulcs), melyet az aláíró az elektronikus aláírás létrehozásához használ.

*Aláírás-ellenőrző adat*: olyan egyedi adat (jellemzően kriptográfiai nyilvános kulcs), melyet az elektronikusan aláírt elektronikus dokumentumot megismerő személy az elektronikus aláírás ellenőrzésére használ.

*Aláírás-létrehozó eszköz*: olyan hardver, illetve szoftver eszköz, melynek segítségével az aláíró az aláírás-létrehozó adatok felhasználásával az elektronikus aláírást létrehozza.

*Aláíró*: az a természetes személy, aki az aláírás-létrehozó eszközt birtokolja és a saját vagy más személy nevében aláírásra jogosult.

### 1.2.2 Az elektronikus aláírás típusai

A törvény három különböző biztonsági szintű elektronikus aláírást különböztet meg: (1) az egyszerű elektronikus aláírást, (2) a fokozott biztonságú elektronikus aláírást és (3) a minősített elektronikus aláírást.

#### 1.2.2.1 Egyszerű elektronikus aláírás

*Egyszerű elektronikus aláírás*, amelyhez különösebb joghatások nem kapcsolódnak. Ilyen pl. az email vagy a dokumentum végére begépelt név.

Ez nem alkalmas arra, hogy a dokumentum szerzőjének személyéről vagy a dokumentum tartalmáról hiteles információt szolgáltasson.

#### 1.2.2.2 Fokozott biztonságú elektronikus aláírás

*Fokozott biztonságú elektronikus aláírás*: olyan elektronikus aláírás, amely
a.) alkalmas az aláíró azonosítására,
b.) egyedülállóan az aláíróhoz köthető,
c.) olyan eszközökkel hozták létre, amelyek kizárólag az aláíró befolyása alatt állnak, és
d.) a dokumentum tartalmához olyan módon kapcsolódik, hogy minden – az aláírás elhelyezését követően a dokumentumon tett – módosítás érzékelhető.

A törvény tehát funkcionális követelményeket támaszt az aláírással szemben, és nem szól e követelmények technikai, eljárási és szervezeti megvalósításáról. Mindazok a technikai, eljárási és szervezeti megoldások felhasználhatók, amelyek kielégítik a fokozott biztonságú elektronikus aláírással szembeni követelményeket.

Lényegében az első részben vázolt technikai folyamatok eredményeként létrejövő elektronikus aláírás ezeket a követelményeket teljesíti.

#### 1.2.2.3 Minősített elektronikus aláírás

*Minősített elektronikus aláírás*: olyan – fokozott biztonságú – elektronikus aláírás, amelyet az aláíró biztonságos aláírás-létrehozó eszközzel hozott létre, és amelynek hitelesítése céljából minősített tanúsítványt bocsátottak ki.

A minősített elektronikus aláírás esetében tehát nem az aláírás funkciói, hanem annak biztonsági feltételei változnak. A funkciók azonosak, hiszen ez a fokozott biztonságú elektronikus aláírás egyik fajtája.

Minősített elektronikus aláírás elhelyezéséhez szükséges eszközöket csak a Nemzeti Média- és Hírközlési Hatóság által minősített hitelesítés-szolgáltatóként nyilvántartásba vett szervezet bocsáthat ki, s csak ilyen szolgáltató használhatja tevékenységével kapcsolatban a „minősített” jelzőt.

##### 1.2.2.3.1 A biztonságos aláírás-létrehozó eszköz

Az Eat 1. számú melléklete a *biztonságos aláírás-létrehozó eszközökkel* (a továbbiakban: BALE) szemben szigorú követelményeket támaszt. Eszerint a BALE megfelelő technikai és eljárási megoldásokkal biztosítja, hogy
a.) az aláírás készítéséhez használt aláírás-létrehozó adat aláírónként biztosan mindig különbözzön, s titkossága kellően biztosított legyen, továbbá
b.) az aktuálisan elérhető technológiával kellő bizonyossággal garantálható, hogy az aláírás készítéséhez használt aláírás-létrehozó adat nem rekonstruálható, megvalósítható annak a jogosulatlan felhasználókkal szembeni védelme, illetve az aláírás nem hamisítható.

Az első követelmény teljesítése általában a BALE jó minőségű véletlenszám generátorán múlik, amely az egyedi magánkulcsot generálja, és magát a kulcsot nem lehet kinyerni a BALE belsejéből. Ez megfelel annak az alapvető elgondolásnak, hogy a BALE személyre szabott eszköz, amelyet az aláíró saját maga kontrollál. A második követelmény úgy teljesül, ha a BALE csak megfelelő kriptográfiai algoritmusokkal hajlandó használni a magánkulcsot.

A BALE-nek nem szabad az aláírandó elektronikus dokumentumot az aláírás elhelyezéséhez szükséges mértéken felül módosítania, illetőleg nem akadályozhatja meg azt, hogy az aláíró a dokumentumot az aláírási elhelyezése előtt megjelenítse. A BALE tehát például a szükséges mértékben kiegészítheti a dokumentumot, de nem cserélheti le egy másik dokumentum lenyomatára.

BALE-nek kizárólag olyan aláíró eszköz és egyéb elektronikus aláírási termék tekinthető, amely rendelkezik az NMHH által, vagy az Európai Unió valamely tagállamában nyilvántartásba vett, tanúsításra jogosult szervezetek által erre a célra kiadott igazolással. E feltétel meglétét az NMHH az aláíró eszköz, illetve a szolgáltató nyilvántartásba vételével egyidejűleg, illetve azt követően is ellenőrzi.

Az EU-ban nyilvántartásba vett tanúsító szervezetek általában csak akkor állítanak ki ilyen igazolást, ha az adott eszköz rendelkezik valamely mértékadó nemzetközi tanúsítással. A BALE-k általában intelligens kártyák, de ez nem kötelező feltétel; USB stick is betölthet ilyen szerepet.

#### 1.2.2.3.2 A minősített tanúsítvány

A minősített tanúsítvány igazolja, hogy a magánkulcshoz tartozó nyilvános kulcs kinek a birtokában van. Ilyen tanúsítványt a hitelesítés-szolgáltató kizárólag személyes regisztráció alapján bocsát ki, és ha a magánkulcs illetéktelen személy befolyása alá került, haladéktalanul visszavonja. A minősített hitelesítés-szolgáltató anyagi felelősséget vállal a tanúsítvánnyal okozott károkért, és felelősségbiztosítással is rendelkezik.

A minősített tanúsítványok az Eat 2. számú melléklete alapján a következő elemeket tartalmazzák:

a.) annak megjelölését, hogy a tanúsítvány minősített tanúsítvány,
b.) a hitelesítés-szolgáltató és székhelyének (ország-) azonosítóját,
c.) az aláíró nevét vagy egy álnevet, ennek jelzésével,
d.) az aláírónak külön jogszabályban, illetve a szolgáltatási szabályzatban, illetőleg az általános szerződési feltételekben meghatározott speciális jellemzőit, a tanúsítvány szándékolt felhasználásától függően,
e.) azt az aláírás-ellenőrző adatot, amely az aláíró által birtokolt aláírást készítő adatnak felel meg,
f.) a tanúsítvány érvényességi idejének kezdetét és végét, valamint azt az időtartamot, ameddig a hitelesítés-szolgáltató a 9. § (7) bekezdés szerinti feladatot a tanúsítvány vonatkozásában ellátja,
g.) a tanúsítvány azonosító kódját,
h.) az adott tanúsítványt kibocsátó hitelesítés-szolgáltató fokozott biztonságú elektronikus aláírását,
i.) a tanúsítvány használhatósági körére vonatkozó esetleges korlátozásokat,
j.) a tanúsítvány felhasználásának korlátait,
k.) más személy (szervezet) képviseletére jogosító elektronikus aláírás tanúsítványa esetén a tanúsítvány ezen minőségét és a képviselt személy (szervezet) adatait.

### 1.2.3 Az elektronikus aláírás joghatásának közös szabályai

Az elektronikus aláírás egyes típusaihoz fűződő jogkövetkezmények a biztonsági szintjüktől függően változnak. Mivel az elektronikus aláírás önmagában csak egy digitális adatsor, melynek feldolgozásához számítógépre és megfelelő célszoftverekre van szükség, így az elektronikus aláírással ellátott dokumentum kinyomtatott változatához nem fűződnek az elektronikus változatra vonatkozó jogkövetkezmények [Eat. 4. § (3)].

#### 1.2.3.1 Közös szabályok

Az azonosítás biztonságától függetlenül nem lehet az elektronikus aláírás, illetve az elektronikus dokumentum elfogadását megtagadni, jognyilatkozat tételére, illetve joghatás kiváltására való alkalmasságát kétségbe vonni kizárólag amiatt, hogy az aláírás, illetve a dokumentum elektronikus formában létezik. [Eat. 3. § (1) bek.] Különösen nem tagadható meg az ilyen aláírás és dokumentum bizonyítási eszközként történő alkalmazása, amivel kapcsolatban az eljáró bíróságnak vagy egyéb hatóságnak az eljárási törvényekben biztosított meglehetősen szabad mérlegelési lehetősége van. E rendelkezésével az Eat. megteremti az *elektronikus írásbeliség* törvényi alapját.

A jelenlegi szabályozás szerint az öröklési jogi és családjogi jogviszonyokban nem lehet a hagyományos papíralapú dokumentumokat mellőzve, kizárólag elektronikus aláírással ellátott elektronikus dokumentumokat felhasználni. Szintén korlátozott az elektronikus dokumentumok és aláírások felhasználásának lehetősége a bírósági eljárásokban és a hatósági jogviszonyokban, de a közigazgatásban egyre több eljárástípusban nyílik meg az elektronikus ügyintézés lehetősége.

Az Eat. a fokozott biztonságú elektronikus aláíráshoz azt a jogkövetkezményt fűzi, hogy az ilyen aláírással ellátott elektronikus irat írásba foglalt iratnak minősül, alkalmazásával elektronikus úton is érvényesen megtehetők az írásbeli alakhoz kötött nyilatkozatok.

Az Eat. kimondja, hogy ha jogszabály a 3. § (2)–(4) bekezdésében foglaltakon kívüli jogviszonyban **írásba foglalást ír elő, e követelménynek eleget tesz** az elektronikusan aláírt elektronikus dokumentumba foglalás is, ha az elektronikus dokumentumot fokozott biztonságú elektronikus aláírással írják alá.

2005 óta a törvény azt is rögzíti, hogy a minősített tanúsítványt bármely bírósági vagy közigazgatási hatósági eljárásban el kell fogadni. [Eat. 3. § (8) bek.]

#### 1.2.3.2 Írásbeli nyilatkozattal kapcsolatos alaki követelmények a régi és az új Ptk-ban

A *régi Ptk*. hatályba lépéséről és végrehajtásáról szóló 1960 évi 11. tvr. 38. §-a (Ptké.) az írásbeliség értelmezése kapcsán úgy rendelkezik, hogy ha jogszabály a szerződés érvényességéhez írásbeli alakot rendel, jogszabály eltérő rendelkezése hiányában írásbeli alakban létrejött szerződésnek kell tekinteni (többek között) a fokozott biztonságú elektronikus aláírással aláírt okirat útján létrejött megegyezést. Ez lényegében megerősíti az Eat. rendelkezéseit a polgári jog területén.

Az új Ptk a kötelmi jogról szóló Hatodik könyv általános rendelkezései között foglalkozik a jognyilatkozatok hatályosságának kérdéseivel, így többek között az írásbeli alak érvényességi kellékeivel. Bár az igen absztrakt szövegezésű Ptk ezen a helyen explicit módon nem utal az elektronikus aláírással ellátott dokumentumnak jognyilatkozat megtétele során történő alkalmazhatóságára, ám a rendelkezések tartalma, a megfogalmazott követelmények összessége egyértelműen lehetővé teszi ezt.

A törvény ugyanis kijelenti, hogy a jognyilatkozat akkor minősül írásba foglaltnak, ha jognyilatkozatát *a nyilatkozó fél aláírta*, továbbá akkor is írásba foglaltnak kell tekinteni a jognyilatkozatot, ha annak közlésére a jognyilatkozatban foglalt *tartalom* változatlan visszaidézésére, a nyilatkozattevő *személyének* és a nyilatkozat megtétele *időpontjának* azonosítására *alkalmas formában* kerül sor. [Ptk 6:7. § (2)-(3) bek.]

Ha sorra vesszük a fenti rendelkezés lényegi elemeit félreérthetetlenül megállapíthatjuk, hogy a *fokozott biztonságú elektronikus aláírás* illetve az *időbélyegzés* kielégíti a jogszabályi követelményeket, hiszen a törvényben valóban az alkalmazott technológiától függetlenül van szó aláírásról, illetve lehetővé teszi a nyilatkozattevő személyének azonosítását, a nyilatkozat tartalmának változatlan formában történő visszaidézését és a keletkezés időpontjának megállapítását.

#### 1.2.3.3 Okirati bizonyítási eszközök és elektronikus aláírás a polgári perrendtartásban

A polgári peres és peren kívüli joggyakorlatban és a közigazgatási szervek munkavégzése során is a leggyakrabban alkalmazott bizonyítási eszközök az okiratok. A polgári perrendtartásról szóló 1952. évi III. törvény [Pp.] gyakorlatilag a jogrendszer egészére kiterjedő érvénnyel határozza meg az okirati bizonyítási eszközök fajtáit és azok érvényességének alaki követelményeit. A kiállító személy jellege szerint megkülönböztethetünk *közokiratokat* és *magánokiratokat*. A magánokiratok további osztályai az *egyszerű magánokiratok* és a *teljes bizonyító erejű magánokiratok*.

Közhatalmi eredete okán a *közokirat* a benne foglaltakat – az ellenkező bizonyításáig – teljesen és hitelesen bizonyítja. Közokirat kiállítható hagyományos és elektronikus úton egyaránt. Eszerint a hagyományos papíralapú dokumentumon kívül közokirat az olyan elektronikus okirat is, amelyet bíróság, közjegyző vagy más hatóság, illetve közigazgatási szerv ügykörén belül, a megszabott alakban állított ki. A közokirat teljesen bizonyítja a benne foglalt intézkedést vagy határozatot, továbbá az okirattal tanúsított adatok és tények valóságát, úgyszintén az okiratban foglalt nyilatkozat megtételét, valamint annak idejét és módját. [Pp 195. § (1) bek.]

A minősített elektronikus aláírásnak kiemelt szerepe van a hagyományos papíralapú közokiratról készített elektronikus másolat hitelesítése során is. Az eredeti papíralapú vagy elektronikus közokiratéval azonos bizonyító ereje van ugyanis annak a közokiratról készített elektronikus okiratnak, amelyet a közokirat kiállítására jogosult szerv ügykörén belül, a megszabott alakban készített el, és amelyen *minősített elektronikus aláírást*, valamint - ha jogszabály így rendelkezik - *időbélyegzőt* helyezett el.

A *magánokiratokhoz* nem fűződik a tartalom valódiságának fenti vélelme. Eszerint a teljes bizonyító erejű magánokirat is csak azt tanúsítja, hogy kiállítója az abban foglalt nyilatkozatot megtette, illetőleg elfogadta, vagy magára kötelezőnek ismerte el, feltéve, hogy az alábbi feltételek valamelyike fennáll:

a.) a kiállító az okiratot sajátkezűleg írta és aláírta;
b.) két tanú az okiraton aláírásával igazolja, hogy a kiállító a nem általa írt okiratot előttük írta alá, vagy aláírását előttük sajátkezű aláírásának ismerte el; az okiraton a tanúk lakóhelyét (címét) is fel kell tüntetni;
c.) a kiállító aláírása vagy kézjegye az okiraton bíróilag vagy közjegyzőileg hitelesítve van;
d.) a gazdálkodó szervezet által üzleti körében kiállított okiratot szabályszerűen aláírták;
e.) ügyvéd (jogtanácsos) az általa készített okirat szabályszerű ellenjegyzésével bizonyítja, hogy a kiállító a nem általa írt okiratot előtte írta alá, vagy aláírását előtte saját kezű aláírásának ismerte el, illetőleg a kiállító minősített elektronikus aláírásával aláírt elektronikus okirat tartalma az ügyvéd által készített elektronikus okiratéval megegyezik;
f.) az elektronikus okiraton kiállítója minősített elektronikus aláírást vagy minősített tanúsítványon alapuló fokozott biztonságú elektronikus aláírást helyezett el. [Pp. 196. § (1) bek.]

Az a magánokirat, amely nem elégíti ki a fenti feltételeket egyszerű magánokiratnak minősül. Bizonyító erejéhez nem fűződik törvényi vélelem, így az egyszerű magánokirat bizonyítékként való felhasználása, határozatot megalapozására való figyelembevétele az eljáró bíróság, egyéb hatóság mérlegelésétől függ.

A magánokirat valódiságát csak akkor kell bizonyítani, ha azt az ellenfél kétségbe vonja, vagy a valódiság bizonyítását a bíróság egyéb okból szükségesnek találja. [Pp. 197. § (1) bek.]

Ha kétség merül fel a legalább fokozott biztonságú elektronikus aláírással ellátott elektronikus okirat aláírójának azonossága, illetve az okirat hamisítatlansága, változatlansága felől, ezek megállapítása érdekében a bíróság megkereséssel fordul az elektronikus aláíráshoz tartozó tanúsítványt kibocsátó hitelesítés-szolgáltatóhoz. Ha pedig a kétség az elektronikus okirathoz kapcsolt időbélyegző által igazolt adatokkal kapcsolatban merül fel, a bíróság az időbélyegzést végző szolgáltatót keresi meg.

### 1.2.4 A minősített elektronikus aláírás joghatása

Fentiekkel összhangban ha az elektronikus dokumentumon minősített elektronikus aláírás szerepel – és az aláírás ellenőrzésének eredményéből más nem következik –, vélelmezni kell, hogy a dokumentum tartalma az aláírás óta nem változott [Eat. 4. § (2)]. Az Eat. tehát ebben az esetben felállítja a hamisítatlanság vélelmét.

A minősített elektronikus aláírásra szigorú szabályok vonatkoznak, így ha érvényes minősített elektronikus aláírással ellátott dokumentummal találkozunk, nyugodtan feltételezhetjük, hogy amikor az aláírást készítették, az aláírás-létrehozó adat kizárólag a tanúsítványban szereplő személy birtokában volt. Így feltételezhetjük, hogy a tanúsítványban szereplő személy írta alá dokumentumot. Ennek megfelelően, az Eat. bizonyító erőt rendel a minősített elektronikus aláíráshoz.

Világosan jelzi a minősített elektronikus aláírás kiemelkedő biztonságát, joghatás kiváltására való alkalmasságát és elfogadottságát a bírósági ügyvitel szabályairól szóló 14/2002. (VIII. 1.) IM rendelet egyik rendelkezése. Ebben ugyanis világosan egymás mellett látható a különbség ugyanazon dokumentum minősített elektronikus aláírást – és időbélyegzőt – tartalmazó és nem tartalmazó változatainak bizonyító ereje között. Ugyanis ha a bíróság jogszabály alapján elektronikus okirat kiadására köteles, és a kiadott elektronikus okiratot a bíróság *minősített elektronikus aláírással* és az elektronikus aláíráson elhelyezett *időbélyegzővel* látja el, az *hiteles másolatnak* minősül. Ha pedig a kiadott elektronikus okirat minősített elektronikus aláírást és időbélyegzőt nem tartalmaz, az nem hiteles másolatnak minősül. [14/2002. (VIII. 1.) IM rend. 15/A. §]

A minősített elektronikus aláírással ellátott elektronikus magánokirat a Pp. szerint teljes bizonyító erejű magánokiratnak minősül, azaz az ellenkező bebizonyításáig teljes bizonyítékul szolgál arra, hogy kiállítója az abban foglalt nyilatkozatot megtette, illetőleg elfogadta, vagy magára kötelezőnek ismerte el [Pp. 196. § (1) f)]. Az ilyen magánokirat valódiságát csak akkor kell bizonyítani, ha azt az ellenfél kétségbe vonja, vagy a valódiság bizonyítását a bíróság szükségesnek találja.

Ugyancsak teljes bizonyító erejű magánokirat az elektronikus ügyvédi ellenjegyzéssel ellátott elektronikus okirat. Ennek az aláírási megoldásnak a legnagyobb gyakorlati jelentősége jelenleg az elektronikus cégeljárásban van.

## 1.3 A hitelesítés-szolgáltató szerepe és jogállása

### 1.3.1 Hitelesítés

A hitelesítés az az eljárás, amelynek során a hitelesítő a kriptográfiai kulcspárt – vagy egyéb aláírás-létrehozó és -ellenőrző eszközt – biztonságosan hozzárendeli egy meghatározott és azonosított személyhez. A hozzárendelésről kiállított igazolás a tanúsítvány [Eat. 2. § 21.]. Elektronikus aláírás esetén a hitelesítést az ún. hitelesítés-szolgáltató végzi.

Hitelesítésre a saját kezű aláírás vonatkozásában is szükség van: léteznie kell egy olyan dokumentumnak, amely a saját kezű aláírást külső fél által hitelesítetten az aláíróhoz, annak arcképéhez rendeli. A személyi okmány egyrészt az arckép feltüntetésével azonosítja az okmány tulajdonosát, másrészt az okmányon szereplő aláírást is hozzárendeli az okmány tulajdonosához. Az azonosítás és a hozzárendelés hitelességét, harmadik személy általi kétségbe vonhatatlanságát az okmány (igazolvány) kiállítójának egyértelmű megbízhatósága adja. (Más esetben pl.: az aláírásnak tanúk, közjegyző vagy ügyvéd általi hitelesítése során a tanúk, a közjegyző vagy az ügyvéd aláírása bizonyítja azt, hogy az aláírás valóban az aláírótól származik).

Az elektronikus aláírás hitelesítésével kapcsolatban egységesen az a szabályozási megoldás alakult ki, hogy a hitelesítés nem (állami) hatósági feladat, hanem piaci szolgáltatás, a hitelesítők nem közigazgatási szervek, hanem vállalkozások. A hitelesítés-szolgáltató „megbízható harmadik félként" bekapcsolódik a felek közötti jogviszonyba: az aláíró által fizetett díj ellenében tanúsítja az aláíró személyazonosságát, és ezért felelősséggel tartozik.

A hitelesítés során a hitelesítés-szolgáltató azonosítja az igénylő személyét (gyakorlatilag személyes megjelenés útján, személyazonosító igazolvánnyal azonosítja) és ez alapján tanúsítványt bocsát ki. A hitelesítés-szolgáltató felelős azért, hogy a tanúsítvány a valóságnak megfelelő adatokat tartalmazzon.

### 1.3.2 A hitelesítés-szolgáltató jogállása

#### 1.3.2.1 Szolgáltatási jogosultság

Mivel a hitelesítésszolgáltatást végző vállalkozásokra a jogalkotó igen komoly bizalmi feladatot ruház, ezért velük szemben egyú*ttal átfogó, a szolgáltatás* biztonsági szintje szerint differenciált szervezeti és működési szabályokat állapít meg,

Fokozott biztonságú elektronikus aláírással kapcsolatos szolgáltatásokat a meghatározott pénzügyi követelmények teljesítése esetén bármely belföldi lakóhelyű vagy belföldön tartózkodási hellyel rendelkező természetes személy, illetve belföldi székhelyű (telephelyű) jogi személy vagy jogi személyiség nélküli szervezet nyújthat. A szolgáltatásnyújtás bejelentési kötelezettséghez kötött, amit a szolgáltató legkésőbb a szolgáltatás elindítása előtt 30 nappal a Nemzeti Hírközlési Hatóság felé köteles teljesíteni [Eat. 7. § (1)].

A minősített szolgáltatások nyújtása a nagy fokú személyi, szakmai, műszaki és pénzügyi megbízhatóságnak a NMHH által lefolytatott minősítési eljárás során történő igazolásához kötött [Eat. 8. § (1)]. Minősített szolgáltatást az a szolgáltató végezhet, amelyik

a.) igazolja, hogy a természetes személy, illetőleg a jogi személy vagy jogi személyiséggel nem rendelkező szervezet vezető tisztségviselője, illetőleg vezetője és alkalmazottai büntetlen előéletűek.
b.) igazolja, hogy a természetes személy, a jogi személy vagy jogi személyiséggel nem rendelkező szervezet vezető tisztségviselője, illetőleg vezetője vagy alkalmazottja a jogszabályban meghatározott szakképesítéssel rendelkezik.
c.) rendelkezik a tevékenység biztonságos folytatásához szükséges pénzügyi háttérrel és felelősségbiztosítással.
d.) biztosítja a tevékenység végzéséhez szükséges, az Eat. mellékleteiben és más jogszabályokban meghatározott szervezeti, biztonsági, eljárási, tájékoztatási követelményeket.

Amennyiben a minősítés során bebizonyosodik, hogy a szolgáltató megfelel a jogszabályokban foglalt követelményeknek, az NMHH a hitelesítés-szolgáltatót, mint minősített tanúsítvány kibocsátására jogosult hitelesítés-szolgáltatót nyilvántartásba veszi. A szolgáltató köteles a működésében bekövetkező változást bejelenteni, és a változással érintett körülményekre vonatkozóan a minősítést újra el kell végezni [Eat. 18. § (2), (4)].

#### 1.3.2.2 A hitelesítés-szolgáltató felelőssége

A minősített szolgáltatást nyújtó hitelesítés-szolgáltató, mint megbízható harmadik fél, felelősséget vállal az elektronikus aláírás és az időbélyegző felhasználásával kapcsolatos körülményekért [Eat. 15. §]. Felelőssége fennáll a vele szerződéses kapcsolatban lévő ügyfelével – az aláíróval – szemben, *a szerződésszegésért viselt felelősség polgári jogi szabályai szerint.*

A törvényben foglalt kötelezettségek megszegése esetén fennáll a felelősség a szolgáltatóval szerződéses jogviszonyban nem álló harmadik személlyel szemben is azokkal a jogügyletekkel kapcsolatban, amelyekben az általa kibocsátott tanúsítványt használták fel. Ez olyan többletgaranciát jelent, ami egyrészt növeli az elektronikus aláírás megbízhatóságát, másrészt biztosítja az aláírással kapcsolatban esetlegesen felmerülő károk megtérülését.

A szolgáltató felelőssége fennáll különösen akkor, ha a kárt
a.) a nem tanúsított aláírási termék felhasználása okozta.
b.) az igénylő nem megfelelő tájékoztatása okozta.
c.) az aláíró személyének, illetve képviseleti jogosultságának nem a törvényben foglaltak szerinti azonosítása okozta.
d.) az okozta, hogy a tanúsítványban foglalt adatok neki felróható okból nem felelnek meg a valóságnak.
e.) az okozta, hogy a hitelesítés-szolgáltatótól származó aláírás-létrehozó adat és aláírás-ellenőrző adat nem egyedi.
f.) az adatkezelési és adatvédelmi szabályok megszegése okozta.
g.) Mentesül a felelősség alól
  i.) A szolgáltató felelőssége kizárólag a minősített szolgáltatásokkal – minősített elektronikus aláírással vagy minősített időbélyegzővel –, illetve az ezekkel ellátott elektronikus dokumentumokkal okozott kárért áll fenn.
  ii.) Mentesül a szolgáltató a felelősség alól – mind az aláíróval, mind harmadik személlyel szemben –, ha a kárt az okozta, hogy az aláíró nem tett eleget a törvényben foglalt tájékoztatási kötelezettségeinek [Eat. 13. §].

#### 1.3.2.3 Állami felügyelet

A hitelesítés-szolgáltatás olyan bizalmi jellegű szolgáltatás, amelyre vonatkozóan szükség van állami felügyeletre. A felügyeleti szervek folyamatosan ellenőrzik a szolgáltatás-nyújtás követelményeinek teljesülését.

Az Eat. szerint a felügyeleti feladatokat a Nemzeti Média és Hírközlési Hatóság látja el, ennek keretében
a.) nyilvántartásba veszi a fokozott biztonságú, illetve a minősített szolgáltatásokat nyújtó szolgáltatókat, valamint az elektronikus aláírási termékek tanúsításra jogosult személyeket (szervezeteket).
b.) a minősítést megelőzően, a bejelentéssel egyidejűleg, illetve azt követően, valamint a szolgáltatók működésének időtartama alatt a törvényben meghatározott eljárásban folyamatosan vizsgálja, illetőleg ellenőrzi, hogy a szolgáltatók megfelelnek-e az e törvény, a felhatalmazása alapján kiadott jogszabályok, a szolgáltatási szabályzat, illetve az általános szerződési feltételek előírásainak.
c.) a szolgáltatóval szembeni követelmények nem teljesítése esetén a törvény szerinti intézkedéseket és szankciókat alkalmazza. [Eat. 17. § (1) bek.]

Az NMHH felügyeleti tevékenysége során a törvényben meghatározott intézkedéseket tehet (21-23. §§). A jogsértés súlyára, gyakoriságára, az okozható vagy okozott kár mértékére, a szolgáltató minősített szolgáltatókénti működésére, valamint a korábbi intézkedésekre figyelemmel a következő intézkedéseket teheti, akár együttesen is:
a.) felhívhatja a szolgáltató figyelmét az a vele szemben meghatározott követelmények betartására.
b.) megtilthatja meghatározott technológiák, illetőleg eljárások alkalmazását.
c.) elrendelheti a korábban kiadott minősített tanúsítványok visszavonását, ha valószínűsíthető, hogy a minősített tanúsítvány valótlan adatot tartalmaz, vagy meghamisították, illetőleg ha a hitelesítés-szolgáltató által a minősített tanúsítványok aláírásához használt aláírás-létrehozó eszköz nem biztonságos.
d.) 100.000 forinttól 10.000.000 forintig terjedő bírságot szabhat ki,
e.) törölheti a szolgáltatót a minősített hitelesítés-szolgáltatók nyilvántartásából, ha a nyilvántartásba vételt meg kellett volna tagadni, vagy ha e törvény, illetőleg a felhatalmazása alapján kiadott jogszabályokban foglalt követelmények teljesítése más módon nem biztosítható.

## 1.4 A tanúsítvány

### 1.4.1 A tanúsítvány fogalma

A tanúsítvány a hitelesítés-szolgáltató által kibocsátott igazolás, amely az aláírás-ellenőrző adatot (nyilvános kulcsot) egy meghatározott személyhez kapcsolja, és igazolja e kapcsolat fennállást, az aláíró személy személyazonosságát vagy valamely más tény fennállását, ideértve a hatósági, hivatali jelleget is.

A tanúsítvány gyakorlatilag a hitelesítésszolgáltató elektronikus aláírásával ellátott elektronikus dokumentum, amely tartalmazza az aláíró aláírásellenőrző adatát – tehát a nyilvános kulcsát – és egyéb, az aláíróra, a hitelesítésszolgáltatóra és a felhasználás feltételeire vonatkozó információkat.

A tanúsítvány kiállítható az aláíró által meghatározott álnévre is [Eat. 9. § (4)]. Ebben az esetben a tanúsítvány azt igazolja, hogy az aláírás az álnév tulajdonosától származik. Tényleges személyazonosságot ez esetben nem igazol. A tanúsítvány kiállítható továbbá olyan céllal is, hogy az az aláírót más személy va*gy szervezet ké*pviseletében történő aláírásra jogosítsa fel.

### 1.4.2 Az attribútum-tanúsítvány

Mivel a gyakorlatban nemcsak az aláíró személyazonosságának, hanem egyéb alanyi minőségének is jelentősége van, esetenként az elektronikus aláírásnak ezeket a tulajdonságokat is meg kell jelenítenie. Az aláíró ilyen megkülönböztető ismérve lehet az, hogy természetes személyként, vagy valamely jogi személy képviseletében jár el és használja fel, bocsátja ki az aláírásával ellátott dokumentumot. Ekként lehet tehát valamely szervezet tagja, munkatársa, gazdálkodó szervezet képviselője, valamely áru vagy szolgáltatás vásárlója, előfizetője, illetve valamely hivatásrend keretében eljáró személy – különösen ügyvéd vagy közjegyző.

Természetes és egyszerű megoldásként kínálkozik, hogy az aláíró speciális szerepeivel összefüggő tulajdonságokat – attribútumokat – az aláíró tanúsítványt kibocsátó hitelesítés-szolgáltató tüntesse fel a tanúsítványban. Ez az opció azonban bizonyos hátrányokkal is együtt jár. Azzal például, hogy ettől fogva mindenki, aki az aláíró tanúsítványt kezeli, egyúttal minden olyan személyes adatról is tudomást szerezhet, amely nem tartozik az érdekkörébe. További jelentős gyakorlati hátrány, hogy ezt követően minden olyan adatnak a megváltozása, amely a tanúsítványban szerepel, egyúttal a tanúsítvány visszavonásával, és új tanúsítvány létrehozásának, kibocsátásának a szükségességével jár együtt, ami természetesen jelentősen megnöveli az elektronikus ügyintézéssel kapcsolatos tranzakciós költségeket.

A fenti gyakorlati nehézségek kiküszöbölésére szolgáló technológiai fejlesztés az attribútum- tanúsítvány. Ennek felhasználása során az aláíró egyes speciális ügyköreihez kapcsolódó tulajdonságokat a hitelesítés-szolgáltató helyett a megfelelő attribútumot kezelő partner igazolja. Ezek az adatok pedig a tanúsítványban nem kerülnek feltüntetésre, hanem csupán az attribútummal kapcsolatos igazolásokat kell a tanúsítványhoz csatolni. A tanúsítvánnyal könnyen összekapcsolható, szabványos formátumú, számítógéppel feldolgozható dokumentumként létrehozott igazolást *attribútum-tanúsítványnak* nevezzük.

Attól függően, hogy egy hitelesítés-szolgáltató milyen információkat, valamint hol és hogyan tüntet fel valakinek a tanúsítványában, az illető jogosultságai más és más módon állapíthatóak meg a tanúsítvány alapján. A szabványosításnak különösen nagy jelentősége van, ugyanis együttműködési, kompatibilitási problémákhoz vezet az, ha a hitelesítés-szolgáltató az egyes információkat a szabványtól eltérően tünteti fel az ügyfél számára kiadott tanúsítványban. Ilyenkor előfordulhat, hogy egy személyazonosítási rendszer nem ismeri fel és nem engedi be az egyébként megfelelő jogosultsággal rendelkező felhasználót, vagy nem az őt megillető jogosultságokat engedélyezi számára, illetve ennek az ellenkezője is megtörténhet, hogy ugyanis jogosulatlan felhasználót enged be.

### 1.4.3 Érvényességi idő

Biztonsági okokból a tanúsítvány érvényességi ideje nem korlátlan. A jogalkotó általános korlátozást a minősített tanúsítványokkal kapcsolatban állapít meg, amelyek érvényességi ideje nem haladhatja meg az aláírás-ellenőrző adat-

hoz kapcsolható aláírás-létrehozó eszközzel összefüggésben meghatározott érvényességi időt, de legfeljebb a kibocsátástól számított két évet [Eat. 35. § (1)].

### 1.4.4 A tanúsítvány közzététele

A hitelesítés-szolgáltató kötelessége a tanúsítványokkal kapcsolatos adatok nyilvántartása, a nyilvántartás folyamatos karbantartása, a változások átvezetése, valamint a nyilvántartásnak közcélú távközlő hálózatok segítségével bárki számára hozzáférhető és folyamatosan elérhető módon közzététele [Eat. 9. § (6)]. Igen jelentős érdek fűződik ahhoz, hogy a tanúsítványok aktuális állapota folyamatosan hozzáférhető legyen. E n*élkül az aláíróval kapcsolatba kerülő fél nem tudhatja, hogy az alkalmazott elektronikus aláírás valóban érvényes-e, azaz nem került-e sor a felfüggesztésére vagy visszavonására, illetve alkalmas-e az adott jogügyeletben történő felhasználásra.*

*A tanúsítvány érvényes a kibocsátástól* számítva az érvényességi idő lejártáig, amikor érvényét veszti. Érvényességi ideje alatt is előfordulhat, hogy a tanúsítványhoz tartozó magánkulcs illetéktelen személy birtokába kerül, ekkor kerül sor a tanúsítvány visszavonására vagy felfüggesztésére.

Kizárólag az érvényes aláíró tanúsítvány alapján készült aláírás az érvényes. Az aláíró tanúsítványának az aláírás pillanatában kell érvényesnek lennie. Ha az aláíró tanúsítványa lejár vagy visszavonásra kerül, a korábban létrehozott aláírások akkor is érvényesek, de ekkor más módon — például időbélyegző segítségével — kell tudni bizonyítani, hogy az aláírás akkor készült, amikor a tanúsítvány még érvényes volt.

### 1.4.5 A tanúsítvány visszavonása

Az elektronikus aláírás csak akkor töltheti be a funkcióját, ha a tanúsítványban szereplő adatok folyamatosan megfelelnek a valóságnak. Ha ezzel kapcsolatban kételyek merülnek fel, akkor a hitelesítés-szolgáltató a tanúsítványt visszavonja, és ezt a tényt a nyilvántartásában haladéktalanul közzéteszi.

A hitelesítés-szolgáltató visszavonja a tanúsítványt akkor is, ha ezt az aláíró, illetve a képviselt személy kéri, vagy ha az NMHH jogerős és végrehajtható határozatában így rendelkezik.

Ha a tanúsítványt kibocsátó hitelesítés-szolgáltató közzéteszi, hogy a tanúsítványt visszavonja, akkor a tanúsítvány már nem érvényes, azt nem szabad elfogadni. A tanúsítványokat általában akkor vonják vissza, ha a hozzájuk tartozó magánkulcs illetéktelen kezekbe került, vagy ha a tanúsítványban szereplő valamely adat megváltozott, vagy ha ezek valamelyike alappal feltételezhető.

A visszavont tanúsítvány többé már nem tehető érvényessé.

### 1.4.6 A tanúsítvány felfüggesztése

A felfüggesztés valójában nem más, mint ideiglenes hatályú visszavonás. A felfüggesztett tanúsítvány éppúgy érvénytelen, mintha visszavonták volna. A különbség az, hogy a felfüggesztett tanúsítványok érvényessége helyreállítható. Tanúsítvány felfüggesztésére általában akkor kerül sor, ha felmerül annak gyanúja, hogy a hozzá tartozó magánkulcs (aláírás létrehozó adat) illetéktelen kezekbe került. ha később bebizonyosodik, hogy a kulcs mégsem volt illetéktelen kezekben, sor kerülhet a felfüggesztett tanúsítvány helyreállítására.

A felfüggesztés a gyors és egyszerű érvénytelenítéshez fűződő lényeges gyakorlati igényt elégíti ki. Mivel azonban a felfüggesztés nem végleges, egy tévesen elrendelt felfüggesztés kisebb problémát okoz, mint egy téves visszavonás. Ugyanakkor a felfüggesztés sok bonyodalmat is okoz: például az előbb felfüggesztett, majd visszaállított tanúsítványok esetén a különböző időpontban begyűjtött visszavonási információk akár eltérő eredményt is adhatnak.

### 1.4.7 Az elektronikus aláíráshoz kapcsolódó egyes szolgáltatások

#### 1.4.7.1 A hitelesítés-szolgáltatás

Ennek keretében a

a.) a hitelesítés-szolgáltató azonosítja az igénylő személyét,
b.) kulcspárt generál az igénylő számára
c.) tanúsítványt bocsát ki, (fogadja a tanúsítványokkal kapcsolatos változások adatait)
d.) nyilvánosságra hozza a tanúsítványhoz tartozó szabályzatokat,
e.) nyilvánosságra hozza az aláírás-ellenőrző adatokat (a nyilvános kulcsot)
f.) nyilvánosságra hozza a tanúsítvány aktuális állapotára (különösen esetleges visszavonására) vonatkozó információkat.

#### 1.4.7.2 Az elektronikus aláíráshoz kapcsolódó egyéb szolgáltatások

Az Eat. hatálya alá tartozó szolgáltatók az elektronikus aláírás hitelesítésén kívül nyújthatnak időbélyegzés-szolgáltatást, valamint aláíráslétrehozó adatnak az aláíráslétrehozó eszközön történő elhelyezésére (pl. a kriptográfiai magánkulcsnak chipkártyán történő rögzítésére) irányuló szolgáltatást.

#### 1.4.7.3 Megszemélyesítés. Aláírás-létrehozó adatnak az aláírás-létrehozó eszközön történő elhelyezése

Az elektronikus aláírás létrehozására általános informatikai célokra használt számítógép, mobiltelefon illetve speciális célhardver egyaránt használható. Az utóbbi eszköz alkalmazása esetén gondoskodni kell az aláírás-létrehozó adatnak a speciális eszközön való elhelyezéséről. Ezzel a művelettel történik meg az aláírás-létrehozó eszköz *megszemélyesítése*. Az Eat. az elektronikus aláíráshoz kapcsolódó szolgáltatásként határozza meg ezt a tevékenységet. [Eat. 16/A. §]

Bár ez a szolgáltatás önálló tevékenységként is ellátható, a magyarországi gyakorlat szerint általában nem erre szakosodott elkülönült eszköz-szolgáltatók nyújtják. Így vagy a hitelesítés-szolgáltatók végzik ezt a tevékenységet is, vagy pedig egyáltalán nem jelenik meg speciális aláíró célhardver, és a magánkulcsot az aláíró fél számítógépe tárolja.

Amennyiben az aláírás-létrehozó és aláírás-ellenőrző adat létrehozását az igénybe vevő számára maga az eszközszolgáltató végzi, úgy kell gondoskodnia a kulcspár létrehozásáról, hogy egyúttal biztosítja az aláíró aláírás-létrehozó adatának (kriptográfiai magánkulcs) titkosságát, valamint az aláírás-ellenőrző adat (kriptográfiai nyilvános kulcs) sértetlenségét. Az aláírás-létrehozó adatot a szolgáltató visszafejtésre alkalmas formában nem tárolhatja, a szolgáltatás nyújtását követően pedig biztosítania kell, hogy az igénybe vevő aláírás-létrehozó adatáról semmilyen másolat ne kerüljön tárolásra.

## 1.5 Időbélyegzés

Egyes esetekben nem csak a dokumentum változatlansága, a küldő fél azonosított személye, hanem a dokumentum keletkezésének vagy módosulásának időpontja is fontos lehet. Ennek megállapítására alkalmas az időbélyegző-szolgáltatás. Az időbélyegző nem tartalmaz információt az aláíró személyére vonatkozóan, hanem az adott dokumentum adott időpontbeli tartalmát igazolja.

A törvény szerint az időbélyegző elektronikusan aláírt elektronikus dokumentumhoz végérvényesen hozzárendelt vagy azzal logikailag összekapcsolt olyan adat, amely igazolja, hogy az elektronikusan aláírt elektronikus dokumentum az időbélyegző elhelyezésének időpontjában változatlan formában létezett. [Eat. 2. § 16.]

Előfordul a gyakorlatban az is, hogy elektronikus aláírással nem rendelkező, műszaki biztonsági jelentőségű elektronikus dokumentumokra kérnek időbélyeget a felhasználók annak későbbi bizonyítása érdekében, hogy a bélyegzés időpontjában a dokumentum már létezett. Ilyen dokumentumok lehetnek jellemzően a hálózati kiszolgáló számítógépek napló állományai (log-file).

### 1.5.1 A minősített időbélyegző bizonyító ereje

Az időbélyegző hitelessége mellett törvényi vélelem áll. Amennyiben ugyanis az időbélyegzést olyan szolgáltató végezte, amely az időbélyegzéskor e szolgáltatás tekintetében a *szolgáltatók nyilvántartásában minősítettként szerepelt* – azaz *minősített időbélyegzés szolgáltató* – és az időbélyegző ellenőrzésének eredményéből más nem következik, az ellenkező bizonyításáig vélelmezni kell, hogy a dokumentumban foglalt adatok az időbélyegző elhelyezése óta változatlan formában léteztek. [Eat. 4. § (6) bek.]

Amennyiben egy dokumentumon elektronikus aláírás, és az aláíráson egy időbélyegző szerepel, akkor nemcsak az bizonyítható, hogy a dokumentumot az a személy írta alá, akinek a tanúsítványa alapján az aláírás ellenőrizhető, hanem az is, hogy a dokumentumot az időbélyegzőben rögzített időpont előtt írta alá, és azóta sem a dokumentum, sem az aláírás nem változott meg.

A minősített időbélyegző bizonyító ereje könnyen felismerhető módon analóg a minősített elektronikus aláírásével. Megjegyzendő ugyanakkor, hogy „fokozott biztonságú elektronikus aláírás" bizonyító erejének nem létezik ekvivalense az időbélyegzők között. Tehát a törvény nem állít fel hasonló bizonyítási vélelmet a „nem minősített időbélyegző" esetére. Bizonyítási eszközként természetesen az ilyen időbélyegző is felhasználható, de bizonyító ereje az eljáró bíróság, hatóság szabad mérlegelésétől függ.

### 1.5.2 Az időbélyegző szolgáltatás

Az időbélyegzőt az időbélyegzés szolgáltató bocsátja ki és helyezi el a dokumentumon. Az időbélyegző tartalmazza az időbélyegzett dokumentum hash-lenyomatát, az időbélyegzés időpontját, és a szolgáltató saját elektronikus aláírását, mint hitelesítő adatot.

Szigorúan véve az időbélyegző an*nyit iga*zol, hogy egy adott lenyomat egy adott időpontban rendelkezésre állt a szolgáltatónál. Az időbélyegző létezéséből – szűken technikai értelemben – nem következik az, hogy létezett is olyan lenyomatú *dokumentum*. Ugyanakkor egy időbélyegzőt csak úgy értelmes felhasználni, ha a hozzá tartozó dokumentum is felmutatható. Az időbélyegzővel csak ekkor igazolható, hogy a szóban forgó – esetleg vitatott hitelességű – dokumentum már létezett a kérdéses időpontban.

Időbélyegző igénylésére illetve kiadására általában elektronikus dokumentumnak legalább fokozott biztonságú elektronikus aláírással történő hitelesítéséhez kapcsolódóan kerül sor. A felhasználó, igénylő – aki általában azonos az elektronikus aláírás felhasználójával – az aláírásra használt aláírás-létrehozó számítógépes program segítségével kapcsolatba lép a megadott időbélyegzés szolgáltatóval. A szolgáltató a saját hiteles és pontos rendszerórája szerinti időpontot, mint időadatot pedig csatolja a létrehozott elektronikus aláíráshoz, annak formátuma szerint meghatározott helyre. Ennek révén a dokumentum további felhasználása esetén hitelesen igazolható, hogy a dokumentum az adott formában az adott időpontban létezett.

Az időbélyegzés-szolgáltatóra a hitelesítés-szolgáltatókhoz hasonló követelmények vonatkoznak. Az időbélyegzés-szolgáltatónak biztonságos informatikai rendszert kell üzemeltetnie, fizikailag biztonságos környezetben, tanúsított kriptográfiai modulokban kell tárolnia kulcsait, meghatározott bizalmi munkakörökkel kell rendelkeznie, és megbízható, auditálható szervezeti környezetet kell kialakítania. Az időbélyegzés-szolgáltató felelősséget vállal a kibocsátott időbélyegzőkért, ezért stabil pénzügyi háttérrel kell rendelkeznie, és ki kell elégítenie a jogszabályokban támasztott pénzügyi, számviteli követelményeket.

Az időbélyegzés-szolgáltató olyan tanúsítvánnyal rendelkezik, amelyről az érintett felek – köztük a többi időbélyegzés-szolgáltató és hitelesítés-szolgáltató felismerhetik, hogy az időbélyegzőt olyan megbízható fél írta alá, akinek elhihetik, hogy az időbélyegzőben feltüntetett lenyomatú dokumentum az időbélyegzőben feltüntetett időpontban már létezett.

# 2. Hiteles másolatkészítés okiratokról. Elektronikus és papíralapú dokumentumok átalakítása

A gyakorlatban sokszor szembesülnek a hivatali munkát végző intézmények azzal a problémával, hogy valamely dokumentum eredetileg hagyományos módon, papíralapú iratként készült el és került kibocsátásra, de ezt az iratot valamely konkrét helyzetben a gyorsabb ügyintézés érdekében elektronikus úton kellene felhasználni olyan módon, hogy közben megőrizze eredeti bizonyító erejét. Természetesen ennek a fordítottjára is bőven akad példa, arra ugyanis, hogy egy elektronikus formában készült hiteles dokumentumról kellene hiteles papíralapú másolatot előállítani. Mind a technológia, mind a kapcsolódó jogi szabályozás számára feladatot jelent ezeknek a helyzeteknek a kezelése.

A kétirányú hiteles konverzióra a hagyományos – papíralapú – és a digitális hitelesítés megoldásai állnak rendelkezésre. Ez utóbbiak esetében elsősorban az elektronikus aláírásnál bevezetett és megvalósított technológiát lehet és kell alkalmazni. Az alkalmazás kereteit és joghatását pedig a vonatkozó jogszabályok állapítják meg.

Ezt a problémakört a jogrendszer két szinten is szabályozza. Alapvető iránymutatásul a polgári perrendtartás [Pp.] néhány rendelkezése szolgál, ezen túlmenően pedig részletesebb szabályozást találunk a papíralapú dokumentumokról elektronikus úton történő másolat készítésének szabályairól szóló 13/2005 IHM rendeletben. A Pp logikája a másolatkészítést illetően is azt diktálja, hogy különbséget tegyen a közokirat és a magánokirat, illetve az ezekről különböző technológiákkal készített másolatok között.

## 2.1 Általános szabályok

### 2.1.1 Másolat készítése közokiratról

#### 2.1.1.1 Nem digitális technológián alapuló másolat készítése papíralapú közokiratról

A perrendtartás szabályai az 1973. január 1-jén hatályba lépett novella óta rendelkeznek az okiratokról technikai úton készített másolatok kiállításának és bizonyító erejének szabályairól. Természetesen az évtizedekkel ezelőtti szabályozás a kor műszaki eszközei alapján még nem számolhatott a digitális másolatkészítés hitelesítés lehetőségével. Az azóta is – néhány kisebb módosítással – hatályban lévő szabályozás az egyéb műszaki vagy vegyi úton előállított olyan másolatokkal foglalkozik, melyek eredetije csak papíralapú okirat l*ehetett*. E*szer*int *fénykép-, film-*, hangfelvételről, illetve az eredeti *okiratról b*ármely adathordozó útján készült okiratról lehet szó. Tipikus, gyakori alkalmazási területe a fénymásolás (xerográfia) útján előállított másolat.

Bizonyító erő tekintetében az ekvivalencia elve érvényesül, tehát az ilyen felvételnek illetve az adathordozó útján készült okiratnak az eredeti papíralapú közokiratéval azonos bizonyító ereje van, amennyiben a felvételt, illetve az adathordozóról az okiratot bíróság, közjegyző vagy más hatóság, továbbá közigazgatási szerv vagy ezek ellenőrzése mellett más szervezet készítette. [Pp. 195. § (2)]

Természetesen a másolatként létrejött okiratnak tartalmaznia kell azokat a meta-adatokat, amelyekből, megállapítható, hogy mely szervezet, hatóság készítette el, illetve ellenőrizte a készítését. Az elektronikus (digitális) aláírással és időbélyegzővel történő hitelesítés követelménye az ilyen hagyományos módszerekkel készített másolatoknál fel sem merül.

#### 2.1.1.2 Elektronikus (digitális) technológián alapuló másolat készítése papíralapú vagy elektronikus közokiratról

A pontosság érdekében rögzítenünk kell, hogy az 2.1.1.1. pontban tárgyalt másolatkészítési eljárások, akkor is ha elektronikus eszköz, például fénymásoló gép felhasználásával járnak, nem tekintendők a Pp szóhasználata szerinti „elektronikus” technológián alapulónak. A törvény szövegéből egyértelműen megállapítható, hogy az „elektronikus okirat” kifejezést a jogalkotó az olyan digitális technológián, digitális adatábrázoláson alapuló dokumentum számára tartja fenn, amely számítógéppel feldolgozható.

A Pp elektronikus – valójában tehát digitális technológián alapuló – másolatkészítésről szóló rendelkezésének hatálya kiterjed mind az eredetileg papíralapú mind pedig az eredetileg elektronikus úton kiadott közokiratokra. Az ezekről készült elektronikus okiratnak – lényegében másolatnak – a szabályok szerint azonos bizonyító ereje van az eredetivel, amennyiben azt

a.) a közokirat kiállítására jogosult szerv,
b.) ügykörén belül,
c.) a megszabott alakban készíti el és
d.) amelyen minősített elektronikus aláírást vagy minősített tanúsítványon alapuló fokozott biztonságú elektronikus aláírást, valamint
e.) időbélyegzőt helyez el. Ez utóbbi feltétel csak akkor szükséges, ha külön jogszabály így rendelkezik. [Pp. 195. § (3) bek.]

A Pp említett rendelkezése szerint az eredeti közokiratéval azonos bizonyító ereje van annak az elektronikus okiratnak is, amelyet

a.) a közokirat kiállítására jogosult külön jogszabályban meghatározott eljárási rend szerint készített el, illetve
b.) amelyet törvény elektronikus közokiratnak nyilvánít.

#### 2.1.1.3 Papír alapú közokirat készítése elektronikus közokiratról

2009 előtt a Pp. csak a papíralapú okiratok elektronikus okirattá történő átalakításával kapcsolatban szabályozta az okiratok bizonyító erejét, és nem volt tekintettel csak az ellenkező irányú átalakítás lehetőségére. Az ebből a hiányosságból fakadó problémák orvoslására irányult a 2008. évi XXX. törvény, amely – egyébként másolatkészítés technikai részleteinek kibontása nélkül – meghatározza az *elektronikus-papír konverzióval* létrejövő papíralapú közokirat bizonyító erejét. A kérdés a gyakorlatban általában úgy merül fel, hogy hitelesnek tekinthető-e az elektronikus okirat kinyomtatásával létrejövő papíralapú dokumentum. A szabályozás alapelve megköveteli, hogy az ilyen másolatot is hasonló formai elemekkel lássák el, mint az egyéb papíralapú okiratokat.

A szabályozás nem öleli fel az okiratok teljes lehetséges körét, ugyanis csak az *elektronikus közokiratokra* és az ezekről készített papíralapú másolatokra vonatkozik, a magánokiratokra azonban nem. Eszerint tehát az eredeti elektronikus közokiratéval azonos bizonyító ereje van annak az *elektronikus közokiratról* készített papíralapú okiratnak, amelyet

a.) a közokirat kiállítására jogosult
b.) ügykörén belül,
c.) a megszabott alakban vagy
d.) külön jogszabályban meghatározott eljárási rend szerint készített el. [Pp. 195. § (4) bek.]

Ugyanez a jogszabályhely további eseteket is említ. Az elektronikus közokiratot kibocsátó illetve az őrzésére hivatott szerv – akár az elektronikus archiválás szolgáltató – által is készíthető, vagy az ellenőrzése alatt is készülhet papíralapú dokumentum az eredetiről, akár úgy is, hogy a másolat alapját az elektronikus közokiratot tároló adathordozóról származó adatok képezik. Az ilyen dokumentum bizonyító ereje is megegyezik az eredeti elektronikus dokumentuméval.

### 2.1.2 Magánokirat átalakítása közokirattá

A perrendtartási törvény igen körültekintően foglalkozik a másolatkészítés kérdéseivel. Így a rendelkezések a közokiratokon kívül tekintettel vannak a magánokiratokról készített másolatokra, azok bizonyító erejére is. Amennyiben ugyanis a magánokiratról a közokirat kiállítására jogosult szerv – például a közjegyző – ügykörén belül eljárva, a megszabott közokirati alakban készít másolatot, akkor az így kiadott közokirat teljesen bizonyítja, hogy annak *tartalma az eredeti okiratéval megegyezik*.

A közokirati alak ebben az esetben tehát csak azt tanúsítja, hogy a kétféle dokumentum tartalma azonos, ám a tartalom valódiságával kapcsolatban nem rendelkezik olyan bizonyító erővel, mint a közokirat kiadására jogosult szervtől primer módon származó közokirat. Tartalom tekintetében a magánokiratról közokirati alakban készített okirat bizonyító ereje megegyezik az eredeti okiratéval, azaz – mint a Pp. 196. § (1) bekezdés megállapítja – csupán azt tanúsítja, hogy kiállítója az abban foglalt nyilatkozatot megtette, illetőleg elfogadta, vagy magára kötelezőnek ismerte el.

A szabályozás megalkotásakor a jogalkotó nyilvánvalóan fő elvként a *technológia-semlegességet* követte, a rendelkezések ugyanis nem hagynak kétséget afelől, hogy azonos bizonyító erő járul valamennyi lehetséges másolatkészítési esethez, azaz a következőkhöz:

*a.) papíralapú* magánokiratról készített *papíralapú* közokirat,
*b.) papíralapú* magánokiratról készített *elektronikus* közokirat,
*c.) elektronikus* magánokiratról készített *papíralapú* közokirat,
*d.) elektronikus* magánokiratról készített *elektronikus* közokirat.

Azokban az esetekben, amikor a átalakítás, másolatkészítés eredményeként elektronikus közokirat áll elő, a hitelesség, a megfelelő bizonyító erő megállapításának feltétele az is, hogy a közokirat kiállítására jogosult az okiratot, a közokiratokról készített másolatok kiadásánál alkalmazott elvhez hasonlóan

a.) minősített elektronikus aláírással vagy
b.) minősített tanúsítványon alapuló fokozott biztonságú elektronikus aláírással és
c.) ha jogszabály így rendelkezik – időbélyegzővel lássa el, vagy
d.) azt külön jogszabály által meghatározott eljárás szerint készítse el. [Pp. 195. § (5) bek.]

## 2.2 Elektronikus dokumentumról papíralapú másolat készítése

Ennek a konverziónak a szabályait a szabályozott elektronikus ügyintézési szolgáltatásokról [SZEÜSZ] és az állam által kötelezően nyújtandó szolgáltatásokról szóló 83/2012. (IV. 21.) Korm. rendelet [a továbbiakban: SZEÜSZ-rendelet] tartalmazza.

Az ilyen átalakítások végrehajtására az Elektronikus Ügyintézési Felügyelet, valamint a Kormány által kijelölt szervezet jogosult. A hiteles átalakítás alapja ebben az esetben a dokumentumhoz kapcsolódó metaadatok rögzítése.

Amennyiben az elektronikus dokumentum papír alapon történő hiteles megjelenítésének műszaki feltételei adottak, a másolatban rögzíteni kell [SZEÜSZ-rendelet 86. §. (2) bek.]

a.) az elektronikus dokumentum szöveges és ábrázolt tartalmát,
b.) záradékban
- az eredeti iratot kiadmányozó személy, valamint a hatóság nevének,
- az aláírás időpontjának szöveges megjelenítését,
- szervezeti aláírással ellátott elektronikus dokumentum esetén a szervezeti aláíráshoz tartozó tanúsítvány szerint az aláírót meghatározó adatokat,
- metaadatként az elektronikus dokumentum azonosítására vagy a másolat készítésére vonatkozó azon adatokat, amelyek a dokumentum szövegéből egyébként nem állapíthatók meg, de a másolatot kérő szempontjából jelentőséggel bírnak, (*Itt jegyezzük meg, hogy ez a rendelkezés meglehetősen homályos tartalmú. A másolatot készítő szervezet az egyes gyakori irattípusokkal kapcsolatban valószínűleg hamarosan kidolgozza a megoldást a rögzíteni szükséges adatok köréről, de a ritkábban előforduló nyilatkozatokkal, dokumentumokkal kapcsolatos bizonytalanság még hosszú időn keresztül fennállhat.*)

c.) „az elektronikus dokumentumban foglaltakkal egyező tartalmú irat" záradék szöveget,
d.) a papíralapú másolat keltezését,
e.) a másolatkészítő szervezet elnevezését
f.) és a másolatkészítésért felelős, a másolat hitelesítésére jogosult személy aláírását és bélyegzőlenyomatát
g.) az iratérvényességi nyilvántartás szabályai szerinti hitelesítését.

Előfordulnak olyan technikai körülmények, hogy az eredeti elektronikus dokumentum teljes tartalma nem ábrázolható papírt alapú dokumentum formájában. Ezekre az esetekre a SZEÜSZ-rendelet bevezet néhány egyszerűsítést.

Ha a papíralapú másolat nem tartalmazza az eredeti elektronikus dokumentum teljes szöveges és ábrázolt tartalmát, akkor a záradékban azt kell feltüntetni, hogy a másolat a készítésének alapjául szolgáló elektronikus dokumentumot mely részében tartalmazza. Ha az elektronikus dokumentum jellemzőire tekintettel arról papíralapú másolat nem készíthető, a másolatot készítő hatóság olyan papíralapú kivonatot készít az elektronikus dokumentumról, amely felhasználásra alkalmas. A másolatot készítő hatóság ilyen esetben az elektronikus dokumentum használatának feltételeivel nem rendelkező hatóság vagy ügyfél számára biztosítja az elektronikus dokumentum megismerését.

Mód van továbbá a hitelesítő elemeket is magában foglaló QR-kód – vagy hasonló technikai megoldás – alkalmazására. Ugyanis a papíralapú másolat úgy is elkészíthető, hogy a másolatkészítő szolgáltató a teljes elektronikus dokumentumot a papíralapú másolaton, az adatok elektronikus leolvashatóságát biztosító kód formájában helyezi el.

E kód alapján a papíralapú másolat hiteles, ha a kód alapján visszaállított eredeti elektronikus dokumentum megegyezik a papíralapú másolattal.

## 2.3 Papír alapú dokumentumról elektronikus másolat készítése

A papír-elektronikus konverzió lebonyolítása technikai magyarázatra és kidolgozott szabályozásra szorul. Ennek részletes szabályait a 13/2005. (X. 27.) IHM rendelet [a továbbiakban: Konverziós rendelet] tartalmazza.

### 2.3.1 Általános rendelkezések

A rendeletben szabályozott eljárás célja, papíralapú dokumentumokból joghatás kiváltására alkalmas elektronikus dokumentumot készíteni. A kívánt dokumentum vagy *tartalmi megfelelés* szerint, vagy *képi megfelelés* szerint azonos az eredetivel. A másolatkészítés során a másolatkészítő biztosítja a papíralapú dokumentum és az elektronikus másolat képi vagy tartalmi megfelelését, és azt, hogy minden – az aláírás elhelyezését követően – az elektronikus másolaton végrehajtott módosítás észlelhető legyen.

#### 2.3.1.1 Tárgyi hatály

A rendelet hatálya kiterjed különösen a közokiratra, az egyszerű és a teljes bizonyító erejű magánokiratra és a számviteli bizonylatra. Ezeken a nevesített típusokon túlmenően azonban a jogszabály alkalmazható lényegében valamennyi papíralapú dokumentumra, azaz a rendelet értelmezésében minden olyan papíron rögzített szövegre, számadatsorra, térképre, tervrajzra, vázlatra, képre vagy más adatra, amely bármely eszköz felhasználásával és bármely eljárással keletkezett.

Helyes értelmezés szerint tehát bármely a hatósági vagy bírósági eljárásban felhasznált papíralapú dokumentumról – tekintet nélkül azok eredeti rendeltetésére, besorolására – e rendelet szabályai szerint készíthető hiteles elektronikus másolat.

#### 2.3.1.2 Az elektronikus másolat

Az *elektronikus másolat* nem más, mint valamely papíralapú dokumentumról a rendelet szabályai szerint készült elektronikus eszköz – jellemzően digitális számítógép – útján feldolgozható adategyüttes, amely az eredetivel képileg vagy tartalmilag megegyezik, s a külön jogszabályban meghatározott joghatás kiváltására alkalmas.

Az elektronikus másolat létrejöttének tehát előfeltétele az eredeti dokumentumnak való tartalmi vagy képi megfelelés. A rendelet értelmezésében a *képi megfelelés* az elektronikus másolat azon tulajdonsága, amely biztosítja a papíralapú dokumentum – a joghatás kiváltása szempontjából lényeges – tartalmi és formai elemeinek megismerhetőségét. A jogszabály szerint képileg megfelelő másolat tehát „úgy néz ki”, mint a papíralapú eredeti dokumentum.

A *tartalmi megfelelés* nem biztosítja a képi megfelelést, az azonos megjelenést, rendelkezik azonban mindazokkal a tartalmi elemekkel, amelyek – a dokumentumhoz kapcsolódó metaadatokkal együttesen – biztosítják a papíralapú dokumentum azon tartalmi elemeinek megismerhetőségét, amelyek a joghatás kiváltása szempontjából lényegesek. A metaadatok – leíró módon – valójában a képi felismerhetőség hiányát pótolják olyan információkkal, melyek az eredeti dokumentum keletkezési körülményire illetve fellelhetőségére utalnak.

Lényeges alapfogalom továbbá a *másolatkészítő rendszer*, amely – nem nagyon meglepő módon – nem más, mint a másolatkészítés során alkalmazott hardver- és szoftver-eszközök összessége. Ettől különbözik a *másolatkészítési rend*, amely lényegében a másolatkészítés eljárási és műszaki feltételeit, valamint a kapcsolódó felelősségi kérdéseket tartalmazó belső szabályzat. A másolatkészítőnek rendelkeznie kell ilyen szabályozással, melyet nyilvánosan, elektronikus úton közzétesz. A másolatkészítéssel megbízott vagy arra feljogosított személyek körét is a belső szabályzatban kell meghatározni.

### 2.3.2 Az elektronikus másolat elkészítése

Az elektronikus másolatot a papíralapú dokumentum digitalizálásával – legelterjedtebben digitális lapolvasó (scanner) alkalmazásával – készíthetjük el. A digitalizált dokumentumot metaadatok elhelyezésével kell kiegészíteni, melyek azt egyértelműen az eredeti papíralapú dokumentumhoz rendelik. Az így előkészített dokumentumra a másolatkészítő záradékot vezet rá, melynek szövege: „Az eredeti papíralapú dokumentummal egyező".

A dokumentumhoz kapcsolódó metaadatok:
a.) a papíralapú dokumentum megnevezése;
b.) a papíralapú dokumentum fizikai méretei;
c.) a másolatkészítő szervezet megnevezése és a másolatkészítő személy neve;
d.) a másolatkészítő rendszer, illetve a másolatkészítési szabályzat pontos megnevezése és verziószáma;
e.) a másolatkészítés ideje
f.) az irányadó másolatkészítési rend elérhetősége. [Konverziós rendelet 4. § (3) bek.]

Ha a papíralapú dokumentum valamely tulajdonságai miatt az elektronikus másolat nem tartalmazza a papíralapú dokumentum teljes tartalmát, a másolatkészítő a metaadatok között azt is feltünteti, hogy a másolat az eredeti dokumentumot mely részében tartalmazza.

Hitelességének biztosítása érdekében a másolaton *szervezeti aláírást* kell elhelyeznie. A szervezeti aláírás olyan, legalább fokozott biztonságú elektronikus aláírás, amelynek aláírója jogi személy vagy közhiteles nyilvántartásban szereplő jogi személyiség nélküli szervezet. Szervezeti aláírás hiányában alkalmazható olyan, legalább fokozott biztonságú elektronikus aláírás is, amelyre vonatkozóan a hitelesítés-szolgáltató kizárja az álnév használatát. Ilyen aláíró tanúsítvány kiadása előtt a hitelesítés-szolgáltatónak biztosítania kell, hogy a regisztráció alapjául szolgáló személyazonosság igazolására alkalmas hatósági igazolványban foglalt névvel betű szerint azonos a tanúsítványba foglalt név.

Gyakori eset, hogy közüzemi szolgáltató szervezetek előfizetőik részére tömegesen bocsátanak ki számviteli bizonylatokat. Ezeket a bizonylatokat – ha eredetileg papíralapú dokumentumként kerültek kiadásra – a számviteli törvényben és a vonatkozó egyéb pénzügyi jogszabályokban meghatározott ideig eredeti formában kell a szolgáltatónak megőriznie. A papíralapú iratok hosszú távú biztonságos megőrzése sokkal költségesebb, mint az elektronikus dokumentumoké. Ezért a szolgáltatók az iratok elektronikus átalakítására és megőrzésére törekszenek. Ilyen esetekben a nagy mennyiségű anyag feldolgozására tekintettel bizonyos könnyítő megoldások alkalmazhatók. Több dokumentumon is elhelyezhető ugyanis egy elektronikus aláírás, illetőleg egy időbélyegző. Ezek a dokumentumok azonban a továbbiakban csak együtt kezelhetők.

A tömeges másolatkészítési igényre tekintettel a szolgáltatók automatikus másolatkészítési rendszert is kialakíthatnak.

### 2.3.3 Automatikus másolatkészítés

A papíralapú dokumentum elektronikus másolata automatikusan akkor készíthető el, ha
a.) a másolatkészítés tanúsítvánnyal rendelkező, zárt rendszerben történik. A tanúsítvány igazolja, hogy a másolatkészítő rendszer külső beavatkozástól mentesen, az eredeti és a másolat összerendelését tekintve garantáltan hibamentesen működik.
b.) a másolatkészítő rendszer megfelelő műszaki és szervezési megoldással biztosítja a másolat olvashatóságát és a mintavételezésen alapuló minőségbiztosítást;
c.) a záradék tartalmazza az automatikus másolatkészítés tényét. [SZEÜSZ-rendelet 87/A. §. (1) bek.]

Kis mennyiségű vagy alkalmi másolatkészítés esetén nincs gyakorlati akadálya annak, hogy a másolatkészítő tüzetes vizsgálattal meggyőződjön a metaadatok helyességéről, valamint az eredeti dokumentum és a másolat tartalmi illetve képi megfelelőségéről. Ipari mennyiségű másolat automatikus készítése esetén azonban a dokumentumok egyenként történő ellenőrzése már komoly gyakorlati nehézségeket okoz. Ilyen esetekben – további könnyítő szabályként – a dokumentumonkénti tartalmi ellenőrzés helyett véletlenszerű mintavételezésen alapuló ellenőrzés is alkalmazható.

Ha jogszabály az eredeti dokumentum megsemmisítését lehetővé teszi, az eredeti dokumentum megőrzését abban az esetben is legalább addig biztosítani kell, amíg a másolat olvashatóságát (megnyithatóságát) a másolatkészítő vagy

a másolatot felhasználó nem ellenőrizte és vissza nem igazolta. Az automatikusan készített másolatot minden esetben szervezeti aláírással és időbélyegzővel kell ellátni.

### 2.3.4 Közokirat elektronikus másolata

Papíralapú közokiratról, papíralapú egyszerű illetve teljes bizonyító erejű magánokiratról a közokirat kiállítására jogosult szervezet hiteles elektronikus másolatot állíthat elő. Az így elkészített elektronikus másolatot hitelesíteni kell, melyre a konverziós rendelet szerint kétféle mód áll rendelkezésre:

a.) olyan elektronikus aláírással látja el, amely aláírás megfelel az elektronikus aláírás közigazgatási használatához kapcsolódó követelményekről és az elektronikus kapcsolattartás egyes szabályairól szóló kormányrendeletben a hatóság nevében dokumentum hitelesítésére alkalmazható elektronikus aláírással szemben meghatározott követelményeknek, [85/2012. (IV. 21.) Korm. rendelet 7. §; 12 §; 14. §] vagy

b.) az elektronikus ügyintézés részletes szabályairól szóló kormányrendeletben megadott egyéb módon hitelesíti. [85/2012. (IV. 21.) Korm. rendelet 22. § (2) bek.]

#### 2.3.4.1 Hitelesítés hatósági elektronikus aláírással

Hitelesítésre csak olyan, legalább fokozott biztonságú *elektronikus aláírás használható, amelyhez az aláírás ellenőrzéséhez felhasználható tanúsítványt*

*a.) az NMHH által nyilvántartásba vett Magyarországon székhellyel rendelkező hitelesítés-szolgáltató* bocsátotta ki, és az aláíró személyazonossága hitelt érdemlően megállapítható, azaz

- a tanúsítvány minősített tanúsítvány, vagy
- regisztrációt megelőzően az igénylőnek személyesen meg kell jelennie a regisztrációt végző szervezet előtt, és
- regisztráció során az igénylő személyazonosságát az általa bemutatott hatósági igazolvány alapján ellenőrzi, és
- regisztráció és a személyazonosság ellenőrzése alapjául szolgáló, rögzítendő adatok helyességét az igénylő nyilatkozatban, saját kezű aláírásával ellátva igazolja.

b.) olyan *belföldinek nem minősülő hitelesítés-szolgáltató* bocsátotta ki, amelynél az aláíró személyazonossága a fentiekkel azonos módon hitelt érdemlően megállapítható.

A másolatkészítő számítógépes rendszere számára kibocsátott, a hatóság nevében történő nyilatkozattétel gépi aláírással történő hitelesítésére, valamint az elektronikus adatkapcsolatokban az egyes számítógépes rendszerek egyértelmű azonosítására jogszabályban meghatározott hatóságok esetében csak olyan elektronikus aláírási tanúsítvány használható, amelyhez a tanúsítványt és az aláírás-létrehozó adatot a kormányzati hitelesítés-szolgáltatás keretében hozták létre.

#### 2.3.4.2 Hitelesítés az elektronikus ügyintézési rendeletben megadott egyéb módon

Az elektronikus dokumentumok hitelesítése az elektronikus ügyintézés keretében alapvetően ötféle módon történhet.

a.) fokozott biztonságú vagy minősített elektronikus aláírással,

b.) az azonosításra visszavezetett dokumentumhitelesítés szabályai szerint,

c.) az iratérvényességi nyilvántartásban történő elhelyezéssel,

d.) kizárólag a hatóság zárt informatikai rendszerében történő felhasználás esetén az ilyen esetekre az informatikáért felelős miniszter által kiadott rendeletben meghatározott módon és feltételekkel, vagy

e.) valamely szabályozott elektronikus ügyintézési szolgáltatás [SZEÜSZ] tekintetében jogszabály által megengedett további hitelesítési mód alkalmazásával.

A fenti hitelesítési módok közül külön magyarázatot kíván az iratérvényességi nyilvántartásban való elhelyezés. Ezt a lehetőséget a szabályozott elektronikus ügyintézési szolgáltatások kialakításának rendszere teremtette meg 2012-ben.

#### 2.3.4.2.1 Hitelesítés elektronikus aláírással

A papíralapú dokumentumról készült elektronikus másolat elektronikus aláírással történő hitelesítése megvalósulhat akár a piaci alapon működő hitelesítés-szolgáltatók által, akár pedig a kormányzati hitelesítés-szolgáltató (GOV CA) által kibocsátott tanúsítvány alapján. A kormányzati hitelesítés-szolgáltatást 2013. november 22 óta a Nemzeti Infokommunikációs Szolgáltató Zrt. nyújtja. A szolgáltatással kapcsolatos dokumentumok, szabályzatok a http://http://hiteles.gov.hu/ weblapon érhetők el. A hitelesítéshez szükséges joghatás csak

a.) fokozott biztonságú vagy
b.) minősített elektronikus aláírással váltható ki.

Az aláírást a másolatkészítő szolgáltató helyezi el a dokumentumon, vagy olyan szolgáltató, akinek az aláíró erre kifejezetten megbízást adott.

#### 2.3.4.2.2 Hitelesítés az Iratérvényességi Nyilvántartásban történő elhelyezéssel

Az iratérvényességi nyilvántartást az egyes, az elektronikus ügyintézéshez kapcsolódó szervezetek kijelöléséről szóló 84/2012. (IV. 21.) Korm. rendelet 4. § j) pontja alapján a Közigazgatási és Elektronikus Közszolgáltatások Központi Hivatala [KEKKH] vezeti. Az iratérvényességi nyilvántartás rendeltetése szerint lehetővé teszi a hatóságok által elkészített és a nyilvántartásban elhelyezett

a.) elektronikus dokumentumok, továbbá
b.) az elektronikus dokumentumokról készített hiteles papíralapú másolatok tartalmának és a dokumentum hitelességének ellenőrzését.

Eszerint tehát az iratérvényességi nyilvántartásban elsősorban elektronikus dokumentumok kerülnek elhelyezésre, valamint azok a – jellemzően elektronikusan tárolt – tartalmi és képi metaadatok, melyek alapján az egyes elektronikus dokumentumokról készített papíralapú másolatok hitelessége megállapítható.

Nem ritkán az is előfordulhat, hogy az iratérvényességi nyilvántartás az elektronikus iratot teljes egészében tartalmazza. Ezt – a kizárólag közérdekű adatot vagy közérdekből nyilvános adatot tartalmazó iratok kivételével – titkosítva tárolja. Az okirat kiállítója köteles elhelyezni az iraton a titkosítás feloldásához szükséges kulcsot, valamint az iratnak a nyilvántartásban történő megtekintéséhez szükséges elérési adatokat. Ilyen esetben az irat hitelesnek tekintendő, ha annak az iratérvényességi nyilvántartás igénybevételével történő ellenőrzése sikeres volt.

Az iratérvényességi nyilvántartásba való bejegyzésre több szervezet jogosult. Elsősorban a dokumentumot létrehozó szervezet helyezheti el az általa létrehozott, kibocsátott dokumentumokat. Továbbá a dokumentumról hiteles másolatot készítő szervezet – SZEÜSZ-szolgáltató – és az irat hitelességéről szóló igazolás kiállítására jogosult egyéb szervezet rendelkezik hozzáférési és bejegyzési jogosultsággal.

Az eredetiről készült másolat hitelességének alapvető kritériuma, hogy a másolat tartalmilag, részleteiben, és ha lehet megjelenésében – tehát „képi" attribútumai alapján – is megfeleljen az eredetinek, s így alkalmas legyen az eredeti dokumentum által megcélzott joghatás kiváltására. Az iratérvényességi nyilvántartásban elhelyezett elektronikus dokumentum és a róla készült papíralapú másolat esetében is a tartalmi egyezés az elsődleges követelmény.

Az iratérvényességi nyilvántartásban szereplő elektronikus dokumentumról készült másolat és az eredeti elektronikus dokumentum egyezésére a következő támpontokat állapítja meg az elektronikus ügyintézési rendelet: [85/2012. (IV. 21.) Korm. 23. §]

*a.) Elektronikus dokumentum esetén*: a hatóság által kézbesített elektronikus irat egyezése megállapítható az iratérvényességi nyilvántartásban elhelyezett elektronikus dokumentummal,
*b.) Papír alapú dokumentum esetén*: a hatóság által kézbesített papíralapú irat tartalmi egyezése megállapítható az iratérvényességi nyilvántartásban elhelyezett elektronikus dokumentummal a
*c.) képi megjelenés alapján*, vagy
*d.) egyedi, hamisítás felderítését segítő adat alapján.* Ilyen egyedi adat lehet különösen a sorszámozott matrica, az iraton szereplő záradék, az irat ellenőrzési adatainak eléréséhez szükséges véletlenszerű hivatkozási kód.
*e.) Papír alapú irat lenyomata esetén:* ha a nyilvántartás elektronikus irat hiteles papíralapú lenyomatát tartalmazza, az egyezés megállapítható
- a papíralapú irat és az eredeti elektronikus irat tartalmi egyezése alapján, vagy

- ha az eredeti elektronikus irat lenyomata egyezik az iratérvényességi nyilvántartásban tárolt lenyomattal, vagy
- az iratérvényességi nyilvántartásban tárolt szöveges tartalomra vonatkozó lenyomat ellenőrzése alapján. (Ez utóbbi eljárás csak akkor alkalmazható, ha az eredeti irat nem érhető el.)

### 2.3.5 Papír alapú okirat elektronikus másolatának kiadása bíróság által

Jogszabály bíróságot is kötelezhet arra, hogy valamely papíralapú okiratról elektronikus másolatot készítsen és adjon ki. E másolat elkészítésére tartalmaz – egyes technikai lépéseket is kiemelő – iránymutatást a bírósági ügyvitel szabályairól szóló fentebb már hivatkozott 14/2002. (VIII. 1.) IM rendelet. [14/2002. (VIII. 1.) IM rend. 15/B. §]

Ha ugyanis a bíróság papíralapú okirat elektronikus másolatának kiadására köteles, a másolatot lapolvasó berendezés – közismert nevén scanner – felhasználásával kell elkészíteni. Az irat digitalizálása során a Konverziós rendeletnek megfelelően a képi és tartalmi megfelelőség kialakítására kell törekedni, valamint rögzíteni kell az irat és a kiadmányozó azonosításához szükséges metaadatokat. Ha a digitális kiadmányt a bíróság minősített elektronikus aláírással és az elektronikus aláíráson elhelyezett időbélyegzővel látja el, az hiteles másolatnak minősül.

# 3. Elektronikus dokumentumok archiválása

A hitelesítés-szolgáltatók archiválási szolgáltatást is nyújthatnak, amely elektronikus dokumentumok hosszabb távú hiteles megőrzésére szolgál. Ennek a szolgáltatásnak – minden újszerűsége mellett – régi korokra visszavezethető, jól ismert előképei vannak, az irattárak illetve a levéltárak. Az elektronikus és a papíralapú dokumentumok megőrzésével kapcsolatos nehézségek számos hasonlóságot mutatnak. A papíralapú aláírásoknál éppúgy, mint elektronikus aláírás esetén fennáll az a probléma, hogy a régi iratok és azokat hitelesítő aláírások valódiságát, megbízhatóságát nem könnyű ellenőrizni.

Kétség esetén a papíralapú okirat aláírásának hitelességét addig viszonylag könnyű ellenőrizni, amíg életben van és írásminta szolgáltatására kötelezhető az a személy, akitől az aláírás származik. Egy elhunyt személytől származó – vitatott eredetiségű – aláírás ellenőrzése azonban már csak azon a módon lehetséges, hogy az írásszakértő a vizsgált aláírást összeveti azokkal az írásmintákkal, amelyeket bizonyíthatóan, még élete során a kérdéses személy szolgáltatott.

## 3.1 Időbélyegzés és érvényességi lánc. A probléma és a megoldás – dióhéjban

Számos gyakorlati probléma merül fel az elektronikus aláírással ellátott elektronikus okiratok hosszú távú hiteles megőrzésével kapcsolatban is. Ennek lényege abban foglalható össze, hogy az elektronikus aláírás érvényessége csak addig bizonyítható, ameddig az aláíró tanúsítványa érvényes, amint azonban a tanúsítvány lejár vagy azt felfüggesztik, a tanúsítványhoz kapcsolódó aláírások hitelessége is megszűnik, illetve megkérdőjelezhetővé válik. Maga az aláírás ténye természetesen nem válik meg nem történtté, de esetleg nehéz lesz bizonyítani, hogy az aláíró valóban aláírta-e az adott dokumentumot.

Az időbélyegzés kiküszöböli – valójában csak bizonyos ideig elodázza – a probléma kiéleződését. Ha ugyanis egy még érvényes aláíráson időbélyeget helyeznek el, akkor az aláírás érvényessége addig bizonyítható, amíg az *időbélyegzés szolgáltató* tanúsítványa érvényes. Ekkor ugyanis igazolható, hogy az aláírás már valóban létezett olyan adott időpontban, amikor az aláíró tanúsítványa még érvényes volt, azaz az aláírás érvényes tanúsítvány szerint készült.

Ha tartósabb, hosszú távú megoldásra van szükség – és az információs társadalomban az elektronikus írásbeliség fejlődése határozottan erre utal – akkor a papíralapú dokumentumokhoz hasonlóan az elektronikus aláírással ellátott elektronikus dokumentumok határozatlan időre szóló hiteles és biztonságos megőrzésére is különleges technológiai megoldásokat, és ehhez igazodó speciális szabályokat kell kialakítani. Mint sejthető, a technológia egyik lényegi elemét az időbélyegzés adja, és szükség van egy új típusú közreműködőre, az *archiválás szolgáltatóra*, akinek feladata az elektronikus dokumentumok hosszú távú biztonságos és hiteles megőrzése.

Az *archiválás szolgáltató* rendszeresen új időbélyegzővel látja el a nála letétbe helyezett elektronikusan aláírt dokumentumokat, így biztosítva az úgynevezett érvényességi lánc folytonosságát.

## 3.2 Biztonságos tárolás és technológiakövetés

Annak érdekében, hogy a dokumentum és a hozzá kapcsolódó aláírás sértetlen maradjon, az archiválás szolgáltató gondoskodik az elektronikus dokumentumok biztonságos tárolásáról. A fizikai és vagyonbiztonság kiterjed a redundáns tárolásra, a hozzáférés-védelemre, a tűzvédelemre, a kibernetikus támadások elleni védelemre, s arra hivatott, hogy megakadályozza a dokumentumok véletlen megsemmisülését, szándékos megsemmisítését, s az illetéktelen hozzáférést.

További járulékos feladat az elektronikus dokumentumok olvashatóságának biztosítása. Ismerve és megtapasztalva az informatikában egymást követő technológiai generációváltások gyors ütemét, okkal és joggal kell a szolgáltatóknak felkészülniük arra, hogy a ma széles körben használt dokumentum-formátumok – file-típusok – már néhány évtizeden belül régiségnek számítanak, s egyáltalán nem lesznek általános használatban azok a hardver-eszközök és azok a szoftverek, amelyek eredetileg megjeleníthetővé tették ezek tartalmát. Az archiválás szolgáltató tehát arról is gondoskodik, hogy a megőrzésre átvett dokumentumok olvashatók maradjanak.

A szolgáltató tehát akkor, amikor vállalja egy bizonyos típusú és verziójú file – és az ebben a file-típusban készült dokumentumok – olvashatóságát, akkor egyúttal arra is kötelezettséget vállal, hogy a szolgáltatás fennállási időtartama alatt lesznek olyan eszközei, amelyekkel az adott típusú és verziójú fájlok tartalmát képes lesz megjeleníteni. Ezek nélkül bizonyos idő elteltével már csak azt lehetne igazolni, hogy valaki aláírt egy digitális jelsorozatot, ám annak tartalma már nem volna többé megállapítható.

A hitelesítési technológia, az elektronikus aláírásra használt kriptográfiai algoritmusok és az ezeken alapuló szoftverek is folyamatosan fejlődnek. A szolgáltatónak arról is gondoskodnia kell, hogy az általa üzemeltett rendszer a rendelkezésre álló technológia legmagasabb szintjén álljon, és folyamatosan nyomon kell követnie a technológiai fejlődést. A már elavult kriptográfiai eljárások helyett újakat kell bevezetnie.

Minden aláíráson, amelynek érvényességét a kriptográfiai algoritmusok elavulása érintheti, célszerű még addig elhelyezni egy új időbélyegzőt, amíg az adott aláírás érvényessége még igazolhatóan fennáll, tehát amíg a kérdéses rejtjelező algoritmus még biztonságosnak tekinthető. Az elektronikus aláírás, tanúsítvány vagy időbélyegző készítésére alkalmazható algoritmusok és rejtjelezési szabványok körét a Nemzeti Média és Hírközlési Hatóság határozza meg.

## 3.3 Az elektronikus archiválás jogi szabályozása

Az elektronikus dokumentumok hiteles megőrzésével kapcsolatos legfontosabb szabályokat az elektronikus aláírásról szól törvény 16/C-16/O §§-ban és a digitális archiválás szabályairól szóló 114/2007. (XII. 29.) GKM rendeletben [a továbbiakban: Archiválási rendelet] találjuk. A törvény a digitális archiválási szolgáltatás kereteit, a szolgáltató jogi helyzetét és felelősségét általános érvénnyel szabályozza, míg a rendelet lényegében a gazdálkodó szervezetek által végzett elektronikus archiválási tevékenység egyes speciális szabályait állapítja meg; szervi hatálya nem terjed ki a közfeladatot ellátó szervezetekre.

Az archiválási rendelet nem kizárólag az elektronikusan aláírt dokumentumokra, hanem általában az elektronikus információ hiteles megőrzésére vonatkozik. Eszerint a megőrzésre kötelezett az előírt megőrzési idő lejártáig folyamatosan köteles biztosítani, hogy az elektronikus dokumentumok tárolása, kezelése olyan módon történjen, amely *kizárja az utólagos módosítás lehetőségét*, valamint védi az elektronikus dokumentumokat a törlés, a megsemmisítés, a véletlen megsemmisülés és sérülés, illetve a jogosulatlan hozzáférés ellen. [Archiválási rendelet 2. § (1) bek.]

A rendelet nemcsak a követelményeket határozza meg, hanem az elfogadható megőrzési módszerek kereteit is megadja. Az alapvető megőrzési követelmények tehát kielégíthetők

a.) legalább fokozott biztonságú elektronikus aláírással ellátott dokumentum esetén olyan *archiválási szolgáltató* igénybevételével, akinek működésére az Eat. szabályai irányadók,
b.) olyan *zárt rendszer* alkalmazásával, amellyel kapcsolatban egy akkreditált tanúsító szervezet igazolja, hogy megfelel az informatikai termékek és rendszerek technológiai biztonsági értékelési követelményeit tartalmazó valamely szabvány vagy nyilvános műszaki követelményrendszer előírásainak.
c.) meghatározott körben olyan *elektronikus adatcsere rendszer* (Electronic Data Interchange - EDI) igénybevételével, amely az utólagos módosítás lehetőségének kizárását a minősített szolgáltató által kibocsátott időbélyegző elhelyezésével biztosítja. [Archiválási rendelet 3. §]

A rendelkezésre álló törvényes lehetőségek közül a szakszerűség és a megbízhatóság szempontjából a legkedvezőbb az archiválási szolgáltató igénybevételével történő megőrzés, amelynek részleteit az Eat. szabályozza 2004 óta.

### 3.3.1 Archiválható dokumentumok

A szolgáltatás igénybevevője a szolgáltató rendelkezésére bocsátja a megőrzésre szánt dokumentumokat vagy azok hash-eljárással képzett lenyomatait.

Kizárólag olyan dokumentumok archiválását látják el a szolgáltatók, amelyek legalább fokozott biztonságú elektronikus aláírással kerültek hitelesítésre. A szolgáltatók archiváló rendszere általában a minősített aláírással ellátott dokumentumokra optimalizált. Ha pedig az igénybevevő fokozott biztonságú aláírást helyezett el az archiválandó okiraton, akkor is a minősített szolgáltatóként nyújtott archiválás szolgáltatást veszi igénybe. Mód van arra is, hogy több dokumentumot közös, úgynevezett „keretaláírással" ellátva nyújtsanak be a szolgáltatóhoz.

Az elektronikus dokumentumok illetve lenyomatok átvételekor a szolgáltató ellenőrzi az elektronikus aláírást, és beszerzi az elektronikus aláírás hosszú távú érvényesítéséhez szükséges további információkat:

a.) a tanúsítvánnyal kapcsolatos információkat,
b.) az aláírás-ellenőrző adatot,
c.) a tanúsítvány aktuális állapotára, visszavonására vonatkozó információkat;
d.) a tanúsítvány kibocsátójának szolgáltatói aláírás-ellenőrző adataira és annak érvényességi állapotára vonatkozó információkat. [Eat. 16/E. §]

### 3.3.2 A szolgáltató főkötelezettségei

A tényleges archiválás azzal veszi kezdetét, hogy miután a hosszú távú megőrzéshez szük*séges információkat bes*zerezte, az archiválás szolgáltató minősített időbélyegzőt – helyesen minősített szolgáltatótól származó időbélyegzőt – helyez el az érvényességi láncon, és erről értesíti az igénybe vevőt. Ezt követően pedig – ebben áll a folyamatos megőrzési feladat és szolgáltatás lényege – a szolgáltatási szabályzatban meghatározott időközönként a szolgáltató újabb és újabb minősített elektronikus aláírást és minősített időbélyegzőt helyez el vagy helyeztet el az érvényességi láncon. [Eat. 16/G. §]

A szolgáltatás fennállása alatt a szolgáltató folyamatosan gondoskodik a rábízott elektronikus dokumentumok és lenyomatok oly módon való megőrzéséről, amely kizárja az utólagos módosítás lehetőségét. Emellett a szolgáltató folyamatosan biztosítja az igénybevevő és az egyéb jogosultak számára a dokumentumhoz való hozzáférést, valamint az elektronikus dokumentumok megjeleníthetőségét, olvashatóságát. A szolgáltató olyan rendszert tart fenn, amely védi az elektronikus dokumentumokat a jogosulatlan hozzáférés, módosítás, törlés vagy megsemmisítés ellen. [Eat. 16/H. §; 16/J. §]

### 3.3.3 A szolgáltatás biztonságosságához fűződő vélelem

Az elektronikus archiváló rendszer biztonságos és szakszerű működését a jogalkotó egy bizonyítási vélelem deklarálásával akceptálja, amely szerint ha az elektronikus dokumentum archiválását minősített szolgáltató végzi, az ellenkező bizonyításáig vélelmezni kell, hogy az elektronikus dokumentumon elhelyezett elektronikus aláírás vagy időbélyegző, illetve az azokhoz kapcsolódó tanúsítvány az aláírás és időbélyegző elhelyezésének időpontjában érvényes volt. [Eat. 4. § (7) bek.]

Ez egyenértékű azzal, hogy ha az aláírt dokumentum archiválását minősített archiválás szolgáltató végzi, akkor a bíróságnak abból kell kiindulnia, hogy a feladatát jól, megbízhatóan és szakszerűen látja el, és a bizonyítási teher arra a félre hárul, aki ezzel ellentéteset állít. Ebben a vélelemben áll különösen az archiválás szolgáltató által nyújtott szolgáltatás előnye az egyéni megőrzési megoldásokkal szemben. Ha ugyanis az elektronikus okirat kibocsátója maga végzi az archiválást, aminek törvényes akadálya egyébként nincs, könnyen előállhat az a helyzet, hogy neki magának kell bizonyítania, hogy az archiválást jól végzi.

### 3.3.4 Bizalmasság

Az elektronikus archiválási szolgáltatás keretében a szolgáltató birtokába jut olyan dokumentumoknak, melyek a kibocsátó üzleti titkait, személyes adatait, védett szellemi alkotására vonatkozó adatokat, vagy egyéb esetleg különösen bizalmas tartalmakat hordoznak. Részletesen indokolni sem szükséges, hogy milyen fontos érdekek fűződnek ahhoz, hogy az így átadott információk bizalmassága meg is maradjon, s az igénybevevő kizárólagos rendelkezési szabadsága ne csorbuljon.

Az archivált elektronikus dokumentumok tartalmát a szolgáltató és alkalmazottai, tehát a vele bármilyen munkavégzésre irányuló jogviszonyban álló személyek csak az igénybe vevő írásbeli engedélyével ismerhetik meg. [Eat. 16/I. §]

A szolgáltató munkatársai tehát nem ismerhetik meg az archivált dokumentumok tartalmát. Az archívum az ott elhelyezett dokumentumokat rejtjelezve tárolja, s a bizalmasság védelme érdekében a dokumentumok visszafejtéséhez szükséges kriptográfiai magánkulcs nincsen a szolgáltató birtokában.

Előfordulnak olyan különleges esetek, hatósági eljárások, melyek érdekében a szolgáltatónak mégis hozzá kell férnie a dokumentumok tartalmához. Ekkor a szolgáltató egy különleges eljárás során, több, bizalmi munkakört betöltő munkatársa jelenlétében állítja vissza a magánkulcsot, és a kulcs használatát követően gondoskodnak a nyílt kulcs megsemmisítéséről.

A szolgáltató a dokumentumokat kizárólag akkor bocsátja harmadik fél rendelkezésére, ha erre az igénybevevő felhatalmazta, vagy ha ezt jogszabály írja elő. Ilyen kivételes adatátadásra kerülhet sor

a.) az elektronikus aláírás felhasználásával elkövetett bűncselekmények felderítése vagy megelőzése céljából,
b.) nemzetbiztonsági érdekből,
c.) a tanúsítvány érvényességét érintő polgári peres, illetve nemperes eljárás során az aláíró személyazonosságát igazoló adatok tekintetében.

Az adatátadás címzettje a fenti esetekben a nyomozó hatóság és a nemzetbiztonsági szolgálatok, illetve a polgári perben az ellenérdekű fél és a bíróság. A nyomozó hatóság és a nemzetbiztonsági szolgálat részére történő adatátadás tényét rögzíteni kell, az adatátadásról azonban a hitelesítés-szolgáltató az aláírót nem tájékoztathatja. [Eat. 11. § (2)-(3) bek.] [3/2005. IHM rendelet 6. § (1) bek.]

### 3.3.5 A szolgáltató felelőssége

Az archiválás *szolgáltatót a veszélyes* üzemekre jellemző objektív felelősség terheli. A szolgáltató vétkességére tekintet nélkül köteles megtéríteni az archivált dokumentumok vagy elektronikus aláírások sérülése esetén az okozott kárt. A technológia sajátosságaiból következik, hogy az archivált dokumentumokban keletkezett bármilyen sérülés kimutatható. A szolgáltató a dokumentumok befogadásáról elektronikus aláírással ellátott tértivevényt küld, amely tartalmazza a dokumentum lenyomatát, így az igénybevevőnek rendelkezésére áll a szükséges bizonyítási eszköz a szolgáltató felelősségének megállapításához. [Eat. 16/M. §]

A minősített szolgáltatók a felelősség érvényesíthetősége, helytállási képességük megerősítése érdekében felelősségbiztosítást kötnek. A biztosítási szerződésnek fedezetet kell garantálnia

a.) az elektronikus aláírással vagy időbélyegzővel, illetve az ezzel ellátott elektronikus dokumentummal, illetőleg az archiválási szolgáltató által őrzött érvényességi lánc vagy az elektronikus dokumentum*ok sérülésével vagy megsemmisülésé*vel szerződésen kívül okozott károkra;
b.) az elektronikus aláírással vagy időbélyegzővel, illetve az ezzel ellátott elektronikus dokumentummal, illetőleg az archiválási szolgáltató által őrzött érvényességi lánc vagy az elektronikus dokumentum*ok sérülésével vagy megsemmisülésé*vel szerződésszegéssel okozott károkra.
[Eat. 3. sz. melléklet h).] [3/2005. IHM rendelet 11. §]

### 3.3.6 A szolgáltató személyében bekövetkezett jogutódlás

Mind a hitelesítés-szolgáltatás, mind az elektronikus archiválás az információs társadalom infrastruktúrájához tartozik, s ennek megfelelően kellő stabilitásához, hosszú távú rendelkezésre állásához fontos közérdek fűződik. Ebből következik az, hogy e szolgáltatók tevékenységének beszüntetése különleges szabályozást kíván. Garantálni kell, hogy az elektronikus archiválási szolgáltató ne tűnhessen el a piacról hirtelen és nyom nélkül. A szabályozott megszüntetés hatósági felügyelet mellett történik, melynek során gondoskodni kell a folyamatban lévő ügyek, az üzleti portfólióban lévő ügyfelek átvételéről, további zökkenőmentes kezeléséről.

A szolgáltatónak megszűnése előtt legalább 60 nappal tájékoztatnia kell az érintett ügyfeleket és a Nemzeti Média- és Hírközlési Hatóságot, és meg kell jelölnie, hogy tevékenységét mely más, vele legalább azonos vagy magasabb biztonsági szintű szolgáltató veszi át. [Eat. 16/N. §; 16/O. §]