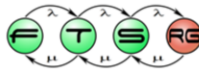


Szolgáltatásbiztonság IT rendszerekben

Micskei Zoltán
(részben Dr. Majzik István előadásai alapján)



Utolsó módosítás: 2015. 04. 22.

Az előadás részben felhasználja a „Szolgáltatásbiztonságra tervezés” MSc tantárgy anyagait, <http://www.inf.mit.bme.hu/edu/courses/szbt>

Szolgáltatásbiztonság??

rendelkezésre állás

hideg tartalék

szoftver redundancia

hibatűrés

nem tervezett leállítás

0 perc leállítás

szoftver RAID

katasztrófa elhárítás

megbízhatóság

HA fürt

visszalépéses helyreállítás

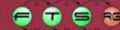
business continuity

replikáció

meghibásodás

„öt kilences” rendszer

hibajavító kódok



Napi informatikai gyakorlatban nagyon sok szót használunk, amik között könnyű elkeveredni. Az előadás megpróbál ezek között kicsit rendet tenni.

Tartalomjegyzék

- **A szolgáltatásbiztonság fogalma**
- A szolgáltatásbiztonságot befolyásoló tényezők
- A szolgáltatásbiztonság eszközei
- Szolgáltatásbiztonság analízise

Szolgáltatásbiztonság

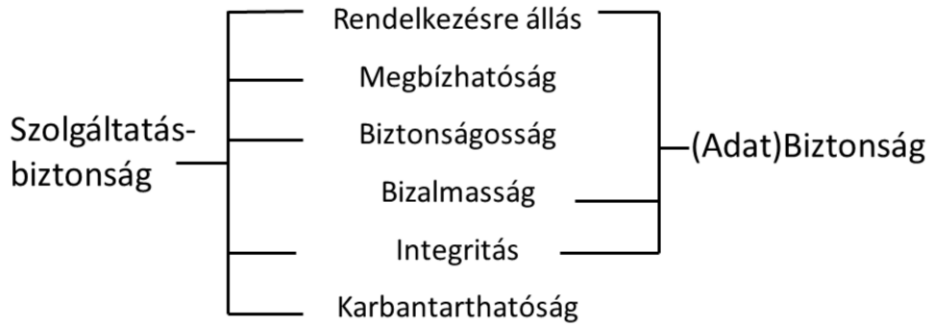
Szolgáltatásbiztonság (dependability):
a képesség, hogy igazoltan bízni lehet a
szolgáltatásban

- *igazoltan*: elemzésen, méréseken alapul
- *bizalom*: szolgáltatás az igényeket kielégíti



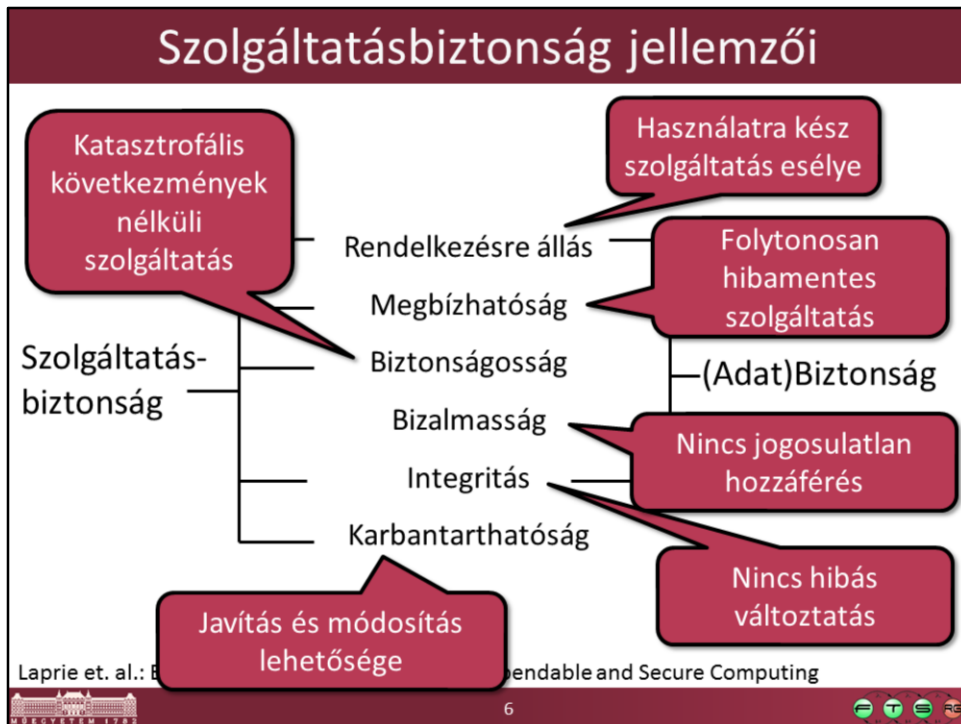
„Dependability is the ability to deliver service that can justifiably be trusted”
(Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, and Carl Landwehr. 2004. Basic
Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Trans.
Dependable Secur. Comput.* 1, 1 (January 2004), 11-33. DOI=10.1109/TDSC.2004.2)

Szolgáltatásbiztonság jellemzői



Laprie et. al.: Basic Concepts and Taxonomy of Dependable and Secure Computing





The original definition of **dependability** is the ability to deliver service that can justifiably be trusted. As developed over the past three decades, dependability is an integrating concept that encompasses the following attributes:

- availability: readiness for correct service.
- reliability: continuity of correct service.
- safety: absence of catastrophic consequences on the user(s) and the environment.
- integrity: absence of improper system alterations.
- maintainability: ability to undergo modifications and repairs.

When addressing **security**, an additional attribute has great prominence, confidentiality, i.e., the absence of unauthorized disclosure of information. Security is a composite of the attributes of confidentiality, integrity, and availability, requiring the concurrent existence of 1) availability for authorized actions only, 2) confidentiality (the absence of unauthorized disclosure of information), and 3) integrity where “improper” alterations means “unauthorized operations”.

Rendelkezésre állás követelményei

Készenléti tényező	Max. kiesés 1 év alatt
2 db 9-es (99%)	3,5 nap
3 db 9-es (99,9%)	9 óra
4 db 9-es (99,99%)	1 óra
5 db 9-es (99,999%)	5 perc
6 db 9-es (99,9999%)	32 másodperc
7 db 9-es (99,99999%)	3 másodperc

Elosztott rendszerek (hibatűrés nélkül, irányadó számok):

- **1 sz gép: 95%**
- **2 sz gép: 90%**
- **5 sz gép: 77%**
- **10 sz gép: 60%**



A következménye ezeknek az, hogy:

- Öt kilences követelmény esetén már az se fér bele általában, hogy a szolgáltatást nyújtó számítógépet újraindítsuk anélkül, hogy más átvonná a szolgáltatás nyújtását, hisz egy újraindítás tovább tart egy szerver esetén 5 percnél.
- Ha több gépből álló rendszerünk van, akkor ott a meghibásodás már egy viszonylag gyakori esemény, és nem csak kivételesen előforduló probléma. Mindenféléképpen kezelni kell valamilyen technikával.

Tartalomjegyzék

- A szolgáltatásbiztonság fogalma
- **A szolgáltatásbiztonságot befolyásoló tényezők**
- A szolgáltatásbiztonság eszközei
- Szolgáltatásbiztonság analízise

Befolyásoló tényezők

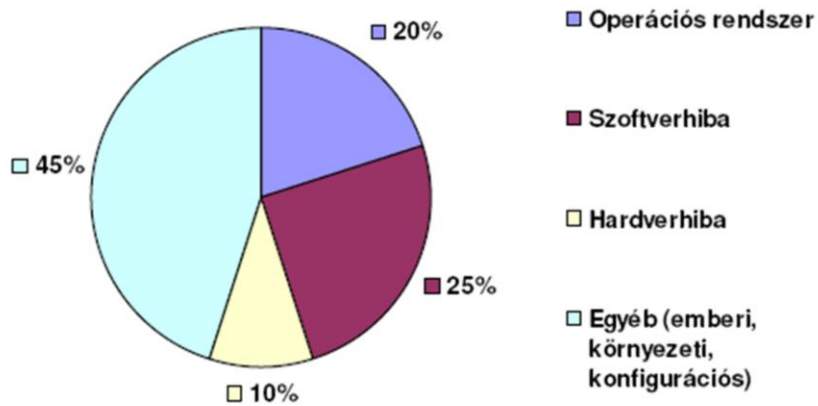
- **Hibajelenség** (failure):
A specifikációnak nem megfelelő szolgáltatás
- **Hiba** (error):
Hibajelenséghez vezető rendszerállapot
- **Meghibásodás** (fault):
A hiba feltételezett oka

Fontos ennek a három fogalomnak a megkülönböztetése

Hatáslánc

- **Meghibásodás → Hiba → Hibajelenség**
 - pl. szoftver:
 - meghibásodás: programozó hiba: csökkentés helyett növel
 - hiba: vezérlés ráfut, változó értéke hibás lesz
 - hibajelenség: számítás végeredménye rossz
 - pl. hardver:
 - meghibásodás: kozmikus sugárzás egy bitet átbillent
 - hiba: hibás memóriacella olvasása
 - hibajelenség: robotkar a falnak ütközik
- **Rendszer hierarchiaszintjének függvénye**
 - alsó szintű hibajelenség felsőbb szinten meghibásodás
 - kimenet beragadás egy chip szintjén hibajelenség
 - rendszer szintjén meghibásodás (chip a cserélhető komponens)

A hibajelenségek okai IT rendszerek esetén



Forrás: Medgyesi Zoltán: Nagy rendelkezésre állású kiszolgálófűrtök vizsgálata, Diplomamunka, BME, 2007.



Több tanulmányt összesítő ábra, természetesen az aktuális számok teljesen eltérőek lehetnek egy adott rendszer esetén. A lényeg csak az, hogy sokféle hibatípusra kell felkészülni, és a legtöbbször a hibatűrés említése kapcsán előkerülő hardver hibák csak ezek kis része.

Meghibásodások kategorizálása

▪ Hardverhibák

- alrendszer (alaplap, processzor, memória)
- tápellátás (tápegység, szünetmentes táp)
- adattároló alrendszer
- hálózat

▪ Szoftverhibák

- az operációs rendszer hibái
- alkalmazáshibák
- illesztőprogram-hibák

▪ ...

▪ Emberi hibák

- rendszergazdai hibák
- illetékes felhasználók nem rosszindulatú hibái
- illetékes felhasználók rosszindulatú hibái
- illetéktelen felhasználók támadásai

▪ Környezeti hatások

- üzemeltetési környezet rendellenességei, például a légkondicionálás leállása, bombariadó, csőtörés
- természeti katasztrófák

Tartalomjegyzék

- A szolgáltatásbiztonság fogalma
- A szolgáltatásbiztonságot befolyásoló tényezők
- **A szolgáltatásbiztonság eszközei**
- Szolgáltatásbiztonság analízise

A szolgáltatásbiztonság eszközei

- **Hiba megelőzés:** Meghibásodás megakadályozása
 - fizikai hibák: jó minőségű alkatrészek, árnyékolás,...
 - tervezési hibák: **verifikáció**
- **Hiba megszüntetés:**
 - prototípus fázis: **tesztelés**, diagnosztika, javítás
 - működés közben: **monitorozás**, javítás
- **Hibatűrés:** Szolgáltatást nyújtani hiba esetén is
 - működés közben: **hibakezelés**, **redundancia**
- **Hiba előrejelzés:** Hibák és hatásuk becslése
 - mérés és „jóslás”, megelőző karbantartás



Fontos tehát, hogy nem pusztán csak a többszörözés az egyetlen módszer, hanem módszerek nagyon széles családja áll a rendelkezésünkre tervezés során.

Hibatűrő rendszerek

Fel kell készülni a **működés közbeni hibákra!**

- **Hibatűrés:** Szolgáltatást nyújtani hiba esetén is
 - működés közbeni autonóm hibakezelés
 - beavatkozás a **meghibásodás** → **hibajelenség** láncba

- **Alapfeltétel: Redundancia (tartalékolás)**
 - többlet erőforrások a hibás komponensek kiváltására

Redundancia megjelenése

1. Hardver redundancia

- többlet hardver erőforrások
 - eleve a rendszerben lévők (elosztott rendszer)
 - hibatűréshez betervezett (tartalék)

2. Szoftver redundancia

- többlet szoftver modulok

3. Információ redundancia

- többlet információ a hibajavítás érdekében
 - hibajavító kódolás (ECC)

4. Idő redundancia

- ismételt végrehajtás, hibakezelés többlet ideje

Együttes megjelenés!

Redundancia típusai

- **Hidegtartalék** (passzív redundancia)
- **Langyos** tartalék
- **Meleg** tartalék (aktív redundancia)



17



Hidegtartalék (passzív redundancia):

normál üzemmódban **passzív**, hiba esetén aktiválva
lassú átkapcsolás (elindítás, állapot frissítés,...)
pl. tartalék számítógép

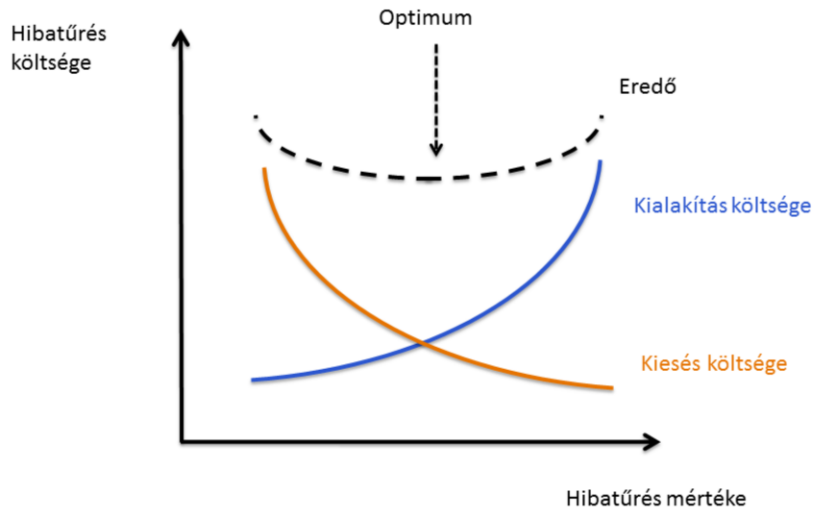
Langyos tartalék:

normál üzemmódban **másodlagos funkciók**
gyorsabb átkapcsolás (indítást nem kell várni)
pl. naplózó gép átveszi a kritikus funkciókat

Meleg tartalék (aktív redundancia):

normál üzemmódban **aktív**, ugyanazt a feladatot végzi
azonnal átkapcsolható
pl. kettőzés, többszörözés

Költségoptimalizálás



A hibatűrés mértékének növelésével nő a kialakítás költsége, de csökken a meghibásodások okozta kár költsége. Általános alkalmazások esetén érdemes valami kompromisszumot keresni.

Hogyan lehet kiszámolni, hogy mennyi a kiesés költsége? Erre jók a következőkben bemutatott technológiák. (Biztonságkritikus alkalmazások esetén nem ilyen egyszerű az optimum megtalálása, pl. mennyi egy emberi élet „költsége”.)

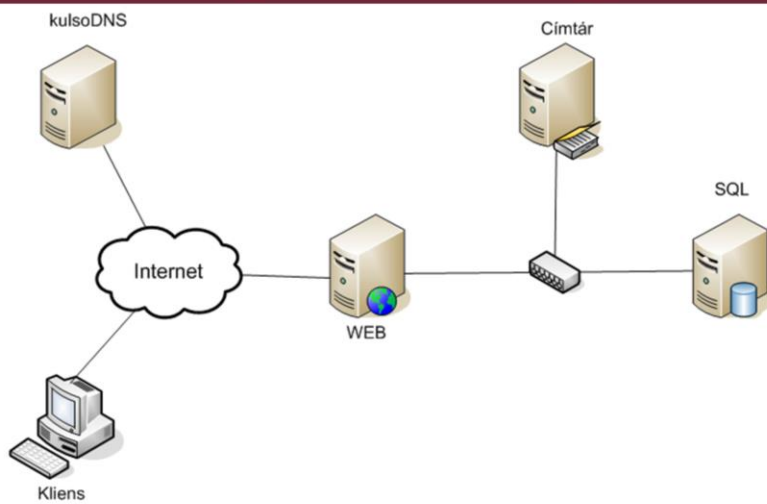
Tartalomjegyzék

- A szolgáltatásbiztonság fogalma
- A szolgáltatásbiztonságot befolyásoló tényezők
- A szolgáltatásbiztonság eszközei
- **Szolgáltatásbiztonság analízise**

Szolgáltatásbiztonság analízise

- Feladatok:
 - Hibamódok, meghibásodások azonosítása
 - Analízis: kvalitatív és kvantitatív
 - ...

Példa: szolgáltatásbiztonság analízise



Feladat: Milyen meghibásodások esetén nem lesz elérhető a szolgáltatás (webáruház)?

Feladat: Meghibásodások azonosítása

- Milyen meghibásodás esetén nem lesz elérhető a szolgáltatás (webáruház)?
- Áramkimaradás, HW hiba, hálózati elem/kábel hiba, szerver szolgáltatások hibája, alkalmazás hiba, frissítés telepítése, túlterhelés, támadás, félrekonfigurálás, verzió inkompatibilitás, vírus...

Hogyan lehetne ezeket
szisztematikusan összegyűjteni?

Hibamód és hatás analízis (FMEA)

- Meghibásodás és hatásaik felsorolása

Komponens	Hibamód	Valószínűség	Hatás
Webszerver	HW hiba	10%	Szolg. kiesés, alkatrész csere
	SW frissítés	80%	Időleges kiesés
SQL szerver	Lemez megtelik	20%	Csak statikus tartalom érhető el
...			

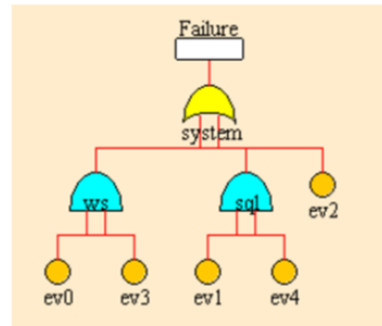


FMEA esetén végigmegyünk a komponenseinken, és megpróbáljuk összeírni, hogy melyeknek milyen hibamódjai vannak és annak mi a következménye. Egyszerű módszer, de már ez is sokat segíthet.

Hibafa (Fault tree)

▪ Hogyan állhat elő a gyökérben lévő hibajelenség?

- legfelső szintű vagy közbenső esemény
- elsődleges (alapszintű) esemény
- ◇ tovább nem vizsgált esemény
- ◡ normál esemény (nem hiba vagy veszély)
- ∩ AND kapu
- ∪ OR kapu



A hibafa arra jó, hogy az egyes meghibásodások közötti kapcsolatokat tudjuk vele könnyen leírni.

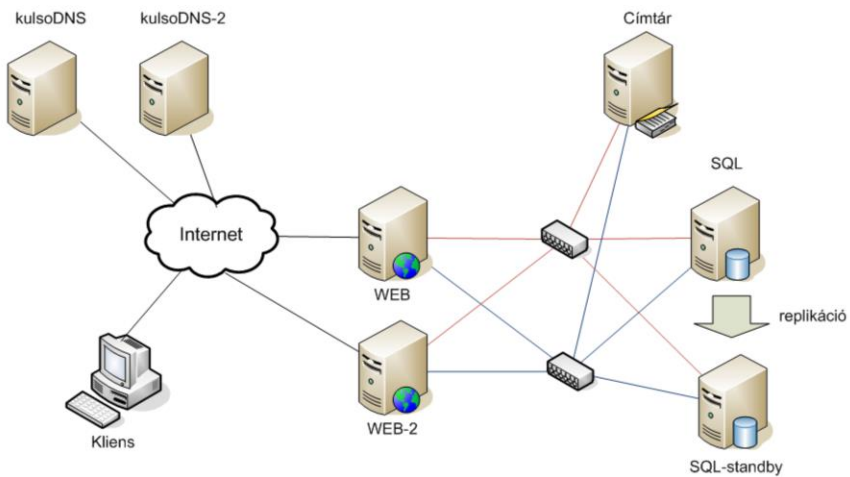
Szolgáltatásbiztonság analízise

- Feladatok:
 - Hibamódok, meghibásodások azonosítása
 - **Analízis: kvalitatív és kvantitatív**
 - ...

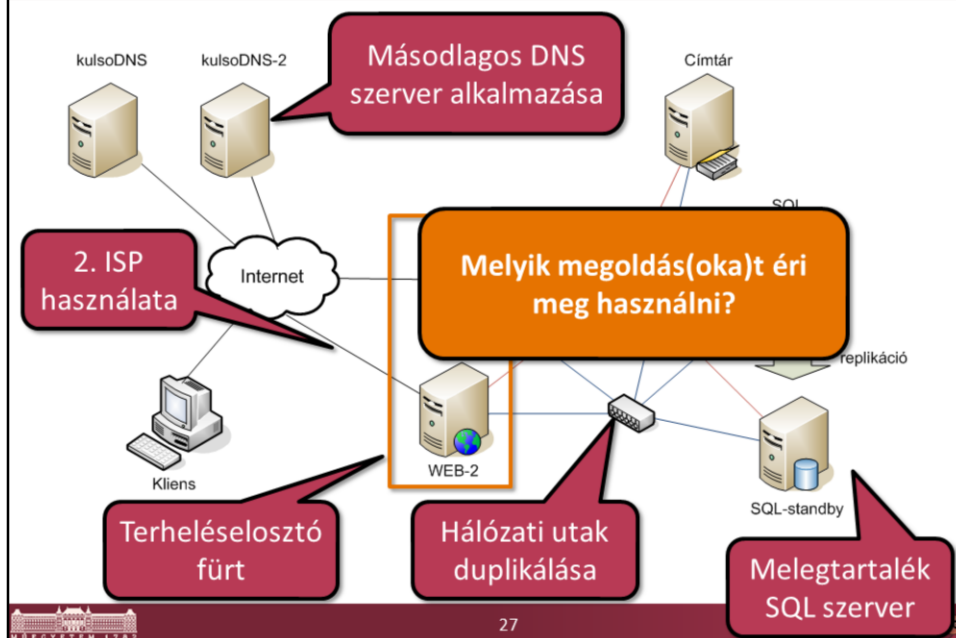
- Módszerek
 - Ellenőrző listák
 - Táblázatok (pl. FMEA: Failure Mode and Effect Analysis)
 - *Hibafák*
 - *Állapot alapú módszerek (pl. Petri hálók)*
 - ...

Megvannak a hibamódjaink, hogyan tudjuk akkor megbecsülni ezekből, hogy a rendszerünk mennyire hibátűrő?

Példa: hibatűrés beépítése

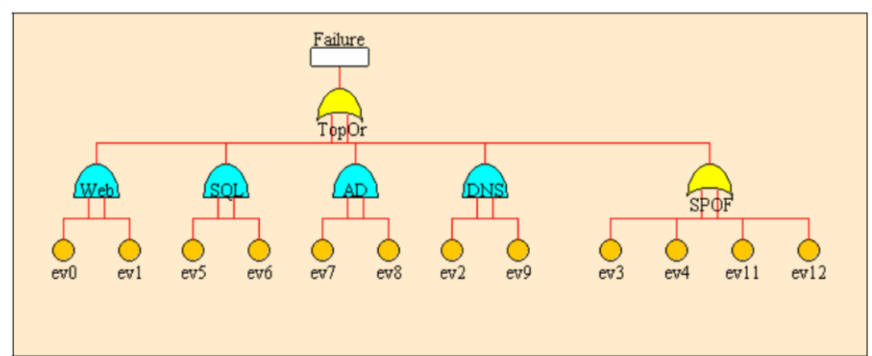


Példa: hibatűrés beépítése



Analízis: hibafa

- [SHARPE](#) eszköz
- Hibafa rajzolása



Hibafa – analízis

- **Kvalitatív:**
 - egyszeres hibapont (SPOF) azonosítása
 - kritikus esemény: több úton is hibajelenséget okoz

- **Kvantitatív:**
 - alapszintű eseményekhez valószínűség rendelése
 - gyökérellem jellemzőjének (pl. megbízhatóság) számolása
 - Probléma: honnan lesznek jó bemenő adataink?

Meghibásodási adatok

- Analízis alapja: meghibásodási valószínűségek
- Honnan lesznek jó adatok:
 - Becslés
 - Saját monitorozó rendszer
 - Külső tanulmányok, számok (híhetőség, pontosság?)
- Példák:
 - Cisco switch MTBF ~ 200000 óra (=22,8 év)
 - IBM S/390 mainframe MTTF 45 év
 - Windows XP MTTF 608 óra
 - webszerver MTTF ~ 16 nap...

Meghibásodási adatok – példa

The Joys of Real Hardware

Typical first year for a new cluster:

- ~0.5 **overheating** (power down most machines in <5 mins, ~1-2 days to recover)
- ~1 **PDU failure** (~500-1000 machines suddenly disappear, ~6 hours to come back)
- ~1 **rack-move** (plenty of warning, ~500-1000 machines powered down, ~6 hours)
- ~1 **network rewiring** (rolling ~5% of machines down over 2-day span)
- ~20 **rack failures** (40-80 machines instantly disappear, 1-6 hours to get back)
- ~5 **racks go wonky** (40-80 machines see 50% packetloss)
- ~8 **network maintenances** (4 might cause ~30-minute random connectivity losses)
- ~12 **router reloads** (takes out DNS and external vips for a couple minutes)
- ~3 **router failures** (have to immediately pull traffic for an hour)
- ~dozens of minor **30-second blips for dns**
- ~1000 **individual machine failures**
- ~thousands of **hard drive failures**
- slow disks, bad memory, misconfigured machines, flaky machines, etc.**

Long distance links: **wild dogs, sharks, dead horses, drunken hunters, etc.**

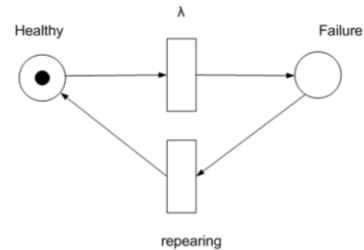
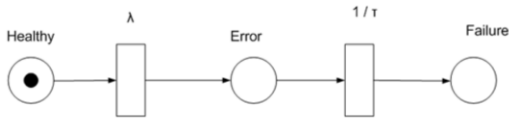
Forrás: Jeff Dean, „Designs, Lessons and Advice from Building Large Distributed Systems”, Google



<http://www.scribd.com/doc/21244790/Google-Designs-Lessons-and-Advice-from-Building-Large-Distributed-Systems>

Időzített Petri hálók

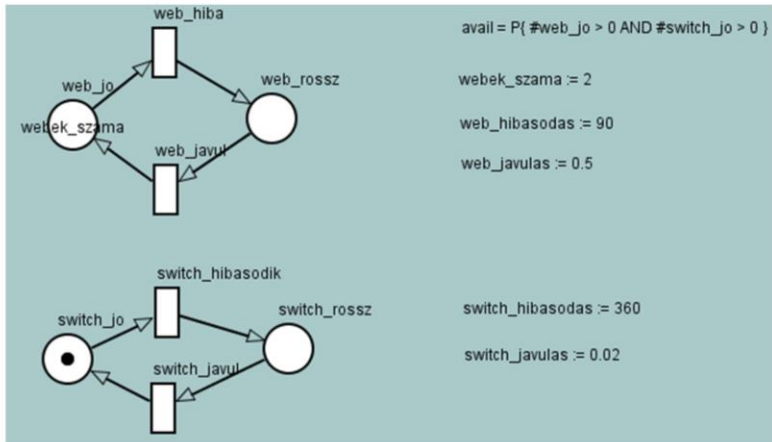
- Elemek:
 - Helyek (kör), tokenek (*Figyelem ez csak most, csak itt jelent állapotot!*)
 - Átmenetek (téglalap)
- Időzítés rendelése az átmenetekhez
 - Determinisztikus
 - Valószínűségi eloszlás alapján
- Alap meghibásodási blokkok:



Fontos! Petri hálónál általában az állapotot a teljes hálón tekintett tokeneloszlás (marking) jelenti. Most szándékosan úgy vesszük fel a hálót, hogy egy komponens állapotát leíró részhálóban pontosan 1 db token legyen, így a helyen lévő token most valóban állapotot jelöl. De általánosan nem igaz az, hogy a (körrel jelölt) hely, mint modellelem állapotot jelöl!

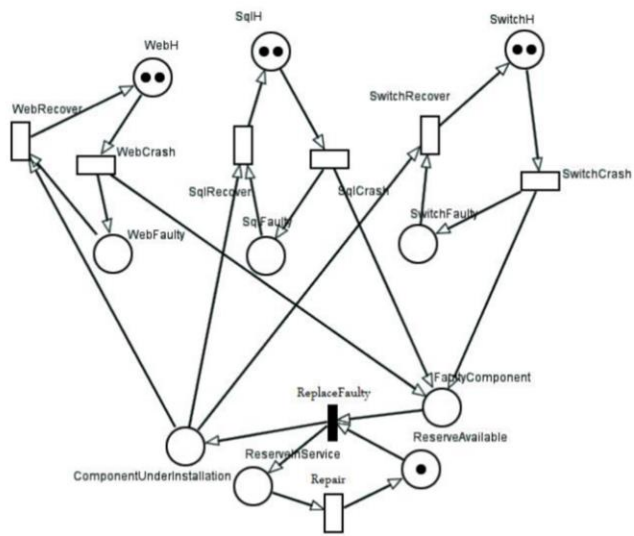
Analízis: Petri-háló

- [TimeNET](#) eszköz
- Alap blokkok és paraméterek



Analízis: Petri-háló

A teljes modell:



Analízis: érzékenység és költségvizsgálat

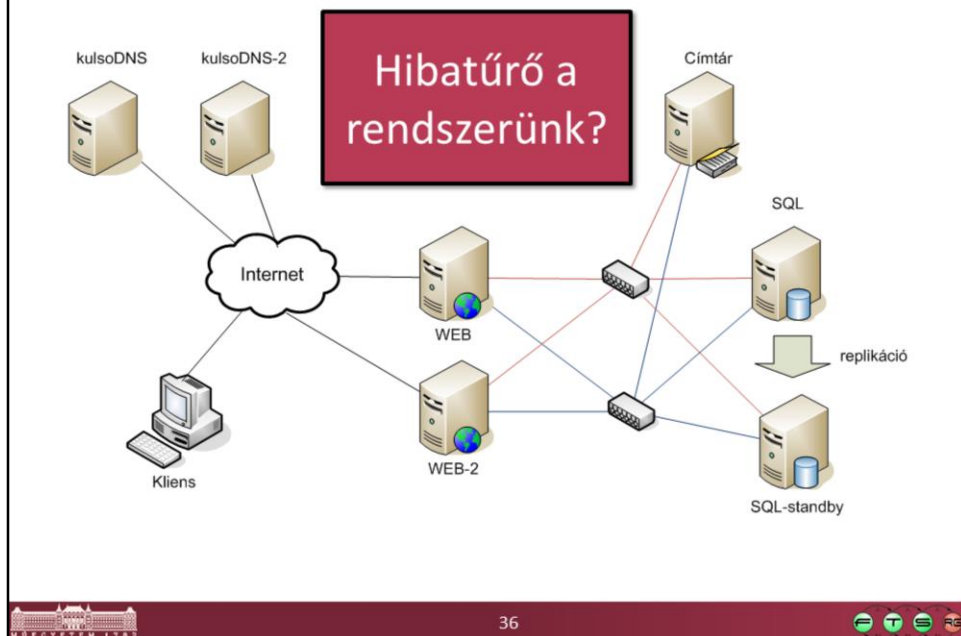
- Érzékenység: melyik paraméter változása befolyásol a legjobban:

ÁTMENET	50% KÉSLELTETÉS	RENDELKEZÉSRE ÁLLÁS	200% KÉSLELTETÉS	RENDELKEZÉSRE ÁLLÁS
WebRecover	0,5	0,8091	2	0,8022
SqlRecover	1	0,8195	4	0,7846
SwitchRecover	0,25	0,8073	1	0,8133
Repair	2	0,9598	8	0,3955

- Költségoptimalizálás:

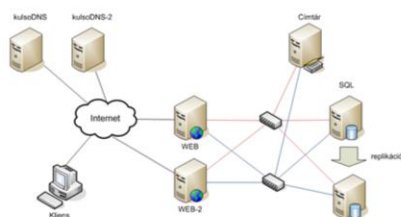
	Költség	Rendelkezésre állás	Kiesés	Kiesés költsége	Nyereség
Alapmodell	0	0,904	34,931	3 493 050,000	
Tartalék SQL	500 000	0,913	31,792	3 179 150,000	-186 100,000
Tartalék web	500 000	0,921	28,945	2 894 450,000	98 600,000
Tartalék mindkettőből	1 000 000	0,930	25,733	2 573 250,000	-80 200,000
Web szerver	1 000 000	0,914	31,463	3 146 300,000	-653 250,000
SQL szerver	2 000 000	0,914	31,536	3 153 600,000	-1 660 550,000
Web szerver + tartalék	2 000 000	0,933	24,565	2 456 450,000	-963 400,000
SQL szerver + tartalék	3 000 000	0,936	23,506	2 350 600,000	-1 857 550,000

Példa: hibatűrés beépítése



Példa: hibatűrés beépítése

Hibatűrő a rendszerünk?



- Attól függ:
 - Bizonyos SPOF-ek ellen védekeztünk
- DE
 - sok kiesési lehetőség maradt még
 - Adatok törlése, teljes serverterem elpusztulása, adminisztrátori hibák, OS hotfix miatti újraindítás...

Példa: hibatűrés beépítése

Tanulság: mindig tudjuk, hogy

- mi ellen akarunk védekezni,
- milyen módszerek vannak arra,
- megéri-e védekezni.

Összefoglalás

- Szolgáltatásbiztonság
 - Jellemzők, hatáslánc, eszközök

- Hibatűrés
 - Redundancia megjelenése

- Analízis:
 - Mérnöki és matematikai módszerek
 - Hibamódok azonosítása
 - Megfelelő védekezési módszer kiválasztása