Haladó adatbiztonság

Oláh Norbert

2022

Összefoglaló

1. IT biztonság és biztonságos kódolás - bevezető óra
   - A sebezhetőségektől a botneteken át a számítógépes bűnözésig
   - A biztonsági hibák osztályozása

IT biztonság és biztonságos kódolás

A sebezhetőségektől a botneteken át a számítógépes bűnözésig

A biztonság jellege

Mi a különbség a security és safety szavak között?

- Biztonsági idővonal
  - A biztonság most azt jelenti, hogy
  - a múltbeli statisztikák alapján mit várunk a jövőben
    - a nemkívánatos események valószínűsége kicsi lesz
    - és / vagy az okozott kár kicsi lesz

Egy meghatározás szerint a security és a safety közötti kritikus különbség a támadó fogalma.
A támadó rosszindulatú, aki szándékosan akar kárt okozni a rendszerekben.
A biztonság célja előrejelezni a jövőt, hogy fel tudjanak rá készülni.

Rumsfeld mondása ”unknown unknowns”:
Link
Biztonsági fenyegetés és a terrorizmus:
Link

Mi az a kockázat?

A kockázat a biztonság negatív mértéke

Definition

A kockázat a veszélyek várható értéke, amelyet egy adott időtartamon belül bekövetkező fenyegetések okoznak
Kifejezhető matematikai képlettel:

$$R = \sum_{t \in T} p_t \cdot d_t$$

R: kockázat
T: Fenyegetések
p_t : A t fenyegetés előfordulásának valószínűsége
d_t : A t fenyegetés előfordulása által okozott kár
Az informatika területén nehéz megbecsülni a kockázatok szintjét.

Ez az egyszerű számítás képezi a kockázatbecslés és a kockázatkezelés alapját számos területen, pl. banki és biztosítási szolgáltatások.
Az informatikai biztonság (IT) területén alkalmazni azonban bonyolult, mivel szinte az összes paraméter bizonytalan.
A bizonytalanság elemei:

- Fenyegetések halmaza (T): végiggondolkodtuk-e az összes lehetséges fenyegetést.
- Valószínűség (pt): attól függ, hogy mennyire kívánatos a cél, milyen könnyű támadni stb.
- Kár (dt): A körülményektől függ, hogy mennyi kárt okoz a támadás.

Tipp: A konkrét értékek helyett az elvont értékeket használják a gyakorlatban mind a valószínűségre, mind a kárra (nagyon alacsony - alacsony - közepes - magas - nagyon magas). A kockázat súlyosságát úgy határozzák meg, hogy e két értéket egy táblázatban összevonják.
Lásd az OWASP link kockázati táblázatát.

OWASP Risk Rating Methodology:
Link

IT biztonság vs. biztonságos kódolás

- Szokásos IT biztonsági technikai követelmények (CIA hármas):
  - Bizalmasság
  - Integritás védelem
  - Elérhetőség
- Kontrollok:
  - Megelőző
  - Detektáló
  - Korrigáló

A biztonságos kódolás a helyes implementációról szól.

- A rendszerre vonatkozó követelmények meghatározásakor figyelembe kell venni a rendszer mely részeit lehet értékesnek tekinteni (potenciális célok) és mely biztonsági követelményeket kell alkalmazni rájuk.
- Ezután a tervezési szakaszban az kontrollt be kell építeni az architektúrába.
- Amikor a biztonságos kódolásról beszélünk, nem elegendő csupán ezeknek az ellenőrzéseknek a biztonságos végrehajtására összpontosítani.
  Bár ez minden bizonnyal fontos része
- A rendszer egészének biztonságos végrehajtására kell összpontosítani - különben egy támadó visszaélhet a biztonsági résekkel, hogy megkerülje a védelmi intézkedéseket.

Lényeg: még a legjobban megtervezett rendszer is sebezhető lehet, ha egyetlen felhasználható kódolási hibát tartalmaz (gyengeség / bug).

OWASP on Controls: Link

A biztonsági hibák jellege

Biztonsági hibák léteztek, léteznek és létezni is fognak ... :)

- Nem lehet teljes mértékben elkerülni (soha nem lehetünk 100% biztosak)
  - Örökkévaló macska és egér játék
  - Vajon érdemes-e sok erőfeszítést fektetni egy ”haszontalan” küzdelembe?
- A teljes védelem garantálása nagyon nehéz feladat
  - Az események 90% -a ismert problémákból származik!
  - A helyes kód írása ”ingyenes”
- Ennek ellenére a jelenlegi helyzet katasztrofális
  - ”Olyan, mintha nagy sebességgel autót vezetne, a biztonsági öv bekapcsolása nélkül.”

- Feltételezhetjük, hogy a kódban nullánál több, jelenleg ismeretlen bug található.
- Néhányan közülük jelen lesznek a biztonsági szempontjából kritikusnak számító kódban
  (például a felhasználói input kezelése vagy biztonsági ellenőrzések)
- Ezek egy részét ki lehet használni az adott eszköz biztonsági követelményeinek megsértése érdekében. Ezt ”sebezhetőségnek” tekintjük.

Sokféle biztonsági kérdés létezik, azonban a **sérülékenységek túlnyomó többségét** a programozás és a tervezés tipikus fejlesztői hibái vagy a biztonsági funkciók félreértése okozzák. *Ezeknek a problémáknak a többségét nem könnyű felismerni, mivel a kód szintaktikailag helyes, és normál körülmények között is helyesen működhet - csak nem azt teszi a program, amit a programozó várt bizonyos esetekben.*

A biztonsági kérdések 95% -a ismert biztonsági résekből származik:
Link

Az SAP azt állítja, hogy az összes támadás 84% -a az alkalmazás rétegén zajlik:
Link

A 2016. évi Veracode-jelentés, amely azt mutatja, hogy az alkalmazások 39% -a nem megfelelő titkosítást használ, és 35% -a beégetett jelszavakat használ:
Link

A nehézség okai

- 1. ok: Ez egy kiegyensúlyozatlan küzdelem
  A fejlesztők rendelkezésre álló ideje és erőforrásai vs. a hackerek motivációja és felkészültsége
- 2. ok: A biztonsági tesztelés kihívást jelent
  A funkcionális tesztelés ellenőrzi, hogy a rendszernek hogyan kellene helyesen működnie, míg biztonság szempontjából arról kell meggyőződnie, hogy a rendszer hogyan nem működik megfelelően.
- 3. ok: Gyenge üzleti motiváció a piaci szereplők által
  A biztonsági szint mérésének technikai nehézségei miatt nincs valódi az ügyfél által kényszerített verseny.
- 4. ok: A végfelhasználók szenvednek a károktól
  A fejlesztők anyagilag nem elég motiváltak

Általában sokkal több hacker van, mint egy adott terméken dolgozó fejlesztő, és ők a világ ”minden idejével” rendelkeznek. A hibák keresése jellemző minden fejlesztési folyamatban, azonban a sebezhetőség tesztelése nagyon más megközelítést igényel.

Noha az első két ok technikai jellegű, a másik kettő piaci kérdés. Tény, hogy a 2000-es évek végén egy vállalat kockázatelemzéssel arra a következtetésre jutott, hogy költséghatékonyabb nem csinálni semmit.

A fertőzött számítógéptől a célzott támadásokig

- Egy átlagos felhasználó azt mondhatja: *”Nem tárolok semmilyen értékes adatot a számítógépemen, szóval miért is érdekelne bárkit?”*
  - Hibás megközelítés
  - Még mindig akarják az Ön erőforrásait (vagy csak az Ön pénzét)
- A hatások globális szinten jelentkeznek - világszerte elterjedt rosszindulatú programok (malware).
  - Közvetlen hatások a felhasználóra: spyware, adware, ransomware,...
  - Botnetek építése - zombi gépek hálózata, amelyet irányítanak.
    - Az Ön erőforrásait széles körű vagy célzott támadások elkövetésére használják (SPAM → phishing, cracking cryptography, DDoS,...)

A fertőzött számítógéptől a célzott támadásokig

- Ezt már nem szórakozásból csinálják
  - Jó pénzt keresnek: cybercrime is a big business
  - Még rosszabb: cyber war and cyber terrorism is felmerülő kérdések
    - Támadnak/hatnak kormányzati helyszíneket, kritikus infrastruktúrákat stb.

A malware (malicious software) olyan szoftver, amelyet a számítógépes műveletek megzavarására, érzékeny információk gyűjtésére, a privát számítógépes rendszerekhez való hozzáférésre vagy a nem kívánt reklám megjelenítésére használnak.
Lásd:
https://en.wikipedia.org/wiki/Malware

Ez a probléma különösen fennáll a tárgyak interneténél (IoT), amely alatt az internethez kapcsolódó (és az interneten keresztül elérhető) eszközöket értjük.

Ezen eszközöknél sokszor hiányoznak vagy rosszul vannak konfigurálva a biztonsági ellenőrzések.
A támadó ezeket felhasználva elérheti a belső hálózatot.

A kiberbűnözés költségei (2021-re várhatóan évente 6T dollár):
Link

A hidegháború alatt az első kiberháborús tevékenység(állítólag) - a CIA által a szovjet gázvezeték szabotálására telepített szoftver-backdoor:
Link

”Zero Days”, egy dokumentumfilm a Stuxnet SCADA-t célzó malware-ről.:
Link

IT biztonság és biztonságos kódolás

A biztonsági hibák osztályozása

A biztonsági hibák osztályozása

- Elméleti taxonómiák (rendszertan) - tudományos megközelítés
  - Risos tanulmány
  - Landwehr's taxonómia
  - Aslam taxonómiák
  - Piessens modellje
  - Weber taxonómia
- Gyakorlati taxonómiák - mindennapi használat
  - 7 Veszedelmes Királyság - Seven Pernicious Kingdoms
  - Plover - Premilinary List of Vulnerability Examples for Researchers
  - CLASP - Comprehensive Lightweight Application Security Process
  - OASIS - WAS (Web Application Security) vulnerability types

A biztonsági hibák osztályozása

- A rendszeresen megjelenő top listák megmutatják a trendeket
  - OWASP Top Ten
  - CWE/SANS Top 25 Legveszélyesebb Szoftver Hibák
  - A szoftverbiztonság 19 halálos bűne - The 19 Deadly Sins of Software Security

Landwehr – A Taxonomy of Computer Program Security Flaws (1994):
Link

Weber és munkatársai – A Software Flaw Taxonomy – Aiming Tools At Security (2005):
Link

Tsipenyuk és munkatársai – Seven Pernicious Kingdoms – A Taxonomy of Software Security Errors (2007):
Link

Christey – PLOVER – Preliminary List of Vulnerability Examples for Researchers (2006):
Link

OWASP Top 10:
Link

CWE/SANS Top 25:
Link

David Le Blanc és munkatársai - 19 Deadly Sins of Software Security:
Link

Landwehr's taxonomy

<table>
<tr><td rowspan="5">Intentional</td><td rowspan="3">Malicious</td><td rowspan="2">Trojan Horse</td><td>Non-replicating</td></tr>
<tr><td>Replicating (virus)</td></tr>
<tr><td colspan="2">Trapdoor</td></tr>
<tr><td colspan="2">Logic/Time Bomb</td></tr>
<tr><td rowspan="3">Non-malicious</td><td rowspan="2">Covert Channel</td><td>Storage</td></tr>
<tr><td>Timing</td></tr>
<tr><td colspan="2">Other</td></tr>
<tr><td rowspan="6">Inadvertent</td><td colspan="3">Validation Error</td></tr>
<tr><td colspan="3">Domain Error</td></tr>
<tr><td colspan="3">Serialization/aliasing (TOCTTOU)</td></tr>
<tr><td colspan="3">Identification/Authentication inadequate</td></tr>
<tr><td colspan="3">Boundary Condition Violation</td></tr>
<tr><td colspan="3">Other Exploitable Logic Error</td></tr>
</table>

A **vírusok** a fájlrendszeren (fájlok, indító szektor) keresztül terjednek, és felhasználói interakcióra van szükség. A moder változatát férgeknek nevezik, amelyek a csatlakoztatott rendszerek és az eszközök sebezhetőségeit kihasználva az interneten keresztül terjednek.
Csapóajtó: ”Backdoor” vagy másnéven egy fejlesztő is beilleszthetti a kódba - ez rámutat arra, hogy a támadók nem feltétlenül külsősök, hanem belsősök is lehetnek (alkalmazottak).
Covert channel: Információkat lehet szerezni úgy, hogy megfigyelünk valamit, amely teljesen kívül esik a tervezett funkción; ezeknek a funkcióknak a feltárása információt hordozhat a ”boxon kívül”.

TOCTTOU

TOCTTOU: Time-of-check-to-time-of-usage.
Ha a támadó beavatkozhat valamilyen erőforrásba (például fájlba) egy apró időablakon belül - az ellenőrzés és az erőforrások felhasználása között - akkor a kód végrehajthatja az utasítást.

- Erre a támadásra jellemző, hogy egy ”symlink” -et tesz a megnyíló fájl helyére, amely egy másik, érzékeny fájlra mutat.
- Ha a ”symlink” az ellenőrzés időpontjában nincs ott, de a használat idején már ott van, a támadó hozzáférhet az érzékeny információkhoz.

Megjegyzés:

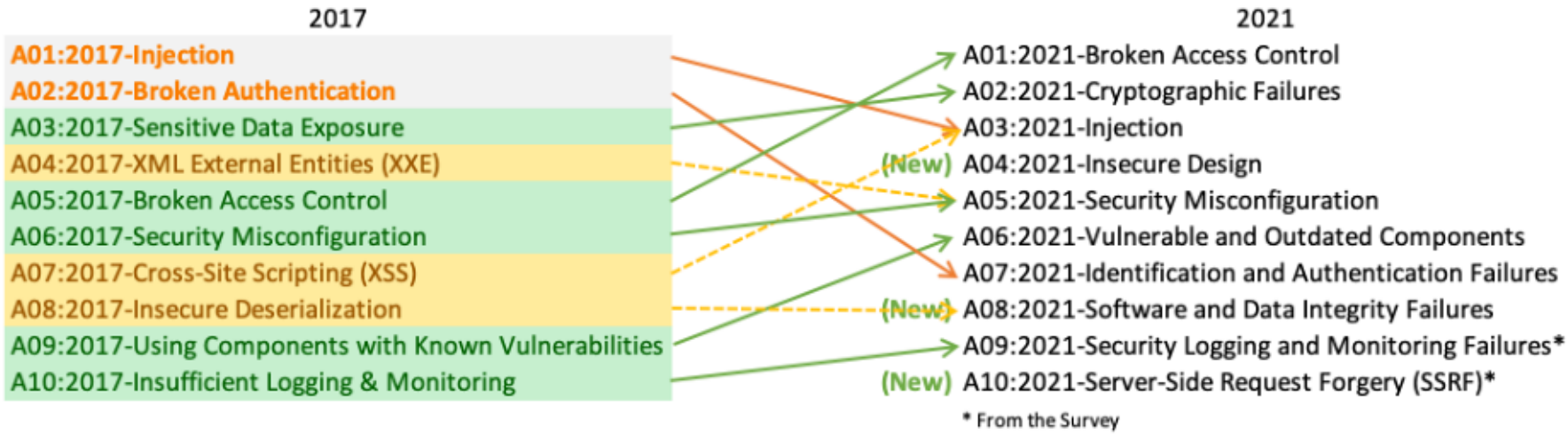
Az időkeretet szerencsével lehert elérni. A támadó szkriptet hoz létre és töröl egy ”symlinket” egy végtelen hurokban (infinity loop), remélve, hogy előbb vagy utóbb megtörténik a megfelelő szinkronizálás.

A hét veszedelmes királyság

- A bemeneti adatok érvényesítése és ábrázolása
  - Bemenet validációjának hibája
- API Abuse / Visszaélés
  - Tipikus hibák az API hívásban
- Biztonsági jellemzők
  - A biztonsági funkciók nem megfelelő használata
- Idő és állapot
  - Problémák az idő és állapotok kezelésével
- Hibák
  - A hibák és kivételek nem megfelelő kezelése
- Kód minőség
  - Rossz minőségű kódból származó veszélyek
- Betokozás -encapsulation
  - Problémák az információk betokozásával
- + Környezet
  - Egyéb hibák, amelyek a környezeti körülményekből és feltételekből származnak

Seven Pernicious Kingdoms - A Taxonomy of Software Security Errors:
Link

OWASP Top Ten 2021



Az első tízben nem csak a sebezhetőségekről van szó. Néhány elem a rendszer egészére kiterjedő kockázatok, mások a program környezetének megfelelő kezelésével foglalkoznak. Ez ”DevOps befolyásnak” tudható be.

OWASP Top Ten 2021 Project:
Link

Egy elemzés a Cydrill-től:
Link

Köszönöm a figyelmet!