A Biztonságos Fejlesztés Tizenhárom Szabálya

„Minden érdekes program tartalmaz legalább egy változót, legalább egy ciklust, és legalább egy hibát” – Murphy

0. Input validálás: ellenőrizni, érvényesíteni (ha kell tisztítani, átformázni, átalakítani)

Gyanakvással kezelj minden külső forrásból származó bemeneti adatot, legyen az parancssorból, hálózati interfészről, környezeti változóból vagy éppen felhasználó által felügyelt fájlból (pl. config) származó adat.

1. Fordítóprogram: figyelj oda a figyelmeztetéseire.

Állítsd magasra a warning szintet (legalább az átadás felé közeledve), és a kód javításával csökkentsd a warning-ok számát.

2. Építés és Tervezés: igazodj a biztonsági eljárásokhoz.

A biztonsági beállítások alkalmazása és kényszerítése legyen része a fejlesztésnek. Tervezéskor vedd figyelembe: ha egy rendszer különböző jogosultságokat igényel adott állapotokban, akkor nem lenne-e egyszerűbb két alkalmazásra bontva fejleszteni, és ezek kommunikációja, ill. saját jogosultságkezelése segít majd az éles üzem biztonságán.

3. KISS: egyszerű a nagyszerű

A tiszta, átlátható és egyszerű kód is segíti a biztonságot, míg a jó kommentelés Neked is hasznos lehet később, amikor a kódhoz kell nyúlni (ha meg hazudsz a kommentekben, akkor a pótolhatatlanságodat segíted, de nincs pótolhatatlan ember).

4. Jogosultságok: alaphelyzet a tiltás.

Jogosultságosztásnál erre alapozz, és ne a kivételekre, vagy a jogosultságok elvételére (oda se add, a „semmi jogot senkinek”-re alapozzál, ne a „minden jogot mindenkinek”-ből vegyél el).

5. Beállítások: igazodj a legkevesebb jogosultság felé.

A „szükséges és elégséges” elve, kiegészítve azzal, hogy időben is csak addig tartson, amíg arra szükség van.

6. Adattisztítás: tisztítsd meg a más rendszer felé küldendő adatokat.

Vigyázz minden olyan adat tisztaságára, melyet összetett alrendszerek felé továbbítasz, mint pl. parancssor, relációs adatbázisok vagy éppen kereskedelmi polcról levehető termékek felé. A támadók kihasználhatják ezek gyengeségeit, és beillesztethetik saját kódjukat a kommunikációba. Az input validálás ekkor nem várható el a fogadó rendszertől, ezért kell a küldőnek gondoskodni róla, hogy az adatok rendben legyenek. A hibaüzenetek is legyenek világosak, ha gond van az input adattal.

7. Védekezés: alkalmazz mélységi és többszintű védekezést.

A többrétegű védekezés hasznos, ha egy(etlen) réteg védelmét megkerülik. A biztonságos fejlesztés a biztonságos üzemeltetéssel vegyítve azt szolgálja, hogy a fejlesztés során nem kiszűrt sebezhetőségeket az éles alkalmazáskor lehessen kiszűrni.

8. Minőség: használj hatékony minőségbiztosítási technikákat.

A jó minőségbiztosítás segít felfedezni és kiküszöbölni a sebezhetőségeket. A különböző típusú tesztek, a kód vizsgálata vagy éppen a külső szakértő (idegen szem mást is észrevehet) alkalmazásával egy biztonságosabb rendszer nyerhető.

9. Szabványok: alkalmazz biztonságos fejlesztés szabványt vagy szabályokat.

A Fejlesztési Módszertanban és a jó gyakorlatokban lefektetett elveket mindenkinek be kell tartania. Közösen alakíthatjuk.

10. Biztonsági szint: határozz meg biztonsági követelményeket.

A tervezés és a fejlesztés korai szakaszaiban dokumentáltan határozd meg a biztonsági elvárásokat, lehetséges fenyegetettségeket, majd győződj meg róla, hogy a megfelelő tudatossággal védekezel ellenük. Terv nélkül megfelelő védelem sincs!

11. Támadások ellen: modellezz lehetséges támadásokat.

Alkalmazz fenyegetettség modellt (pl. PreDeCo/CIA mátrix, Támadási fa) a várható támadásokra. Egységes rendszerben felmért, értékelt és kezelt támadások esetén hatékony védekezés (tervezés – fejlesztés – tesztelés – üzemeltetésben egyaránt).

12. Bizalmasság, sértetlenség: védd az információ épségét és hitelességét

A hibajelzések ne adjanak támpontokat a támadók további lépéseihez. A bizalmas információk tárolása és továbbítása titkosított módon történjen, szabványos és hosszú ideje működő megoldásokkal. Figyeld a konzisztenciát, észleld és jelezd, ha jogosulatlanul változnak adatok vagy beállítások.

Felhasznált forrás: https://www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices (röviden: http://href.hu/x/ikzp)