Információbiztonsági oktatás

Fejlesztők részére

2025

Tartalom

1. IT biztonsági alapismeretek

1.1. Az információbiztonság alapjai

Mi a biztonság?

Az adatbiztonság az információs rendszereken biztonságos módon tárolt, továbbított vagy feldolgozott adatokat jelenti.

Az informatikai biztonság az informatikai rendszerek azon kedvező állapota, amelyben a kezelt adatok bizalmassága (confidentiality), sértetlensége (integrity) és rendelkezésre állása (availability) a rendszer elemeinek szempontjából biztosított (ez az úgynevezett „CIA elv”):

- *bizalmasság:* az információhoz csak az arra jogosultak férhetnek hozzá;
- *sértetlenség:* az információ formája és tartalma az elvárttal megegyező, és egyértelműen azonosítható az információval kapcsolatos műveletek végzője;
- *rendelkezésre állás:* az az állapot, mikor egy informatikai rendszer szolgáltatásai az arra jogosultak számára egy meghatározott időben elérhetőek, és a rendszer elvárt működése sem átmentileg, sem tartósan nem akadályoztatott.

Az információbiztonság jóval tágabb fogalom, mint az IT biztonság, hiszen beletartozik az információ minden megjelenési formája (nem csak elektronikus), valamint a szolgáltatások biztonsága is.

Az állami és önkormányzati szervek elektronikus információbiztonságáról szóló 2013. évi L. törvény (Ibtv.) fő célja, hogy keretet biztosítson az államigazgatási szervek, létfontosságú rendszerelemek, helyi önkormányzatok védelme terén.

1.2. Jogszabályi alapok

DORA rendelet

A Digital Operational Resilience Act (DORA) egy EU-s rendelet, amely 2023. január 16-án lépett hatályba, és 2025. január 17-től alkalmazandó. Célja a pénzügyi szektor informatikai biztonságának megerősítése, hogy a pénzügyi intézmények ellenállóak maradjanak súlyos működési zavarok esetén.

Az alábbi pillérek biztosítják a pénzügyi szektor digitális operatív ellenálló képességének növelését:

1. **ICT kockázatkezelés:** A pénzügyi intézményeknek átfogó keretrendszert kell kialakítaniuk az informatikai kockázatok azonosítására, értékelésére, kezelésére és mérséklésére.
2. **Incidensjelentés**: Mechanizmusokat kell létrehozni a jelentős informatikai incidensek időben történő jelentésére a szabályozó hatóságok felé, beleértve az incidensek részletes dokumentálását és elemzését.
3. **Digitális operatív ellenálló képesség tesztelése**: Rendszeres tesztelést kell végezni a digitális operatív ellenálló képesség biztosítása érdekében, beleértve az alap- és haladó teszteket is.
4. **Harmadik fél kockázatkezelése**: A pénzügyi intézményeknek kezelniük kell a harmadik fél szolgáltatókkal kapcsolatos kockázatokat, beleértve a kulcsfontosságú szerződéses rendelkezéseket is.
5. **Információmegosztás**: A kiberveszélyekkel kapcsolatos információk és hírszerzés cseréje a pénzügyi szektor szereplői között.

MNB 8/2020 ajánlás az informatikai rendszer védelméről

Az ajánlás célja, hogy a pénzügyi közvetítőrendszer tagjai számára gyakorlati útmutatást adjon informatikai rendszerük védelmének kockázatokkal arányos kialakításban, valamint azok védelmére vonatkozó jogszabályi rendelkezések alkalmazásának egységes értelmezésében. Az ajánlás a közösségi és publikus felhőszolgáltatás igénybevételéről szóló 4/2019. (IV. 1.) MNB ajánlásban foglaltakkal, valamint az elektronikus úton megkötött írásbeli szerződésekről, megtett írásbeli jognyilatkozatokról szóló vezetői körlevéllel, a belső védelmi vonalak kialakításáról és működtetéséről, a pénzügyi szervezetek irányítási és kontroll funkcióiról szóló 12/2022. (VIII.11.) MNB ajánlással, valamint a külső szolgáltatók igénybevételéről szóló MNB ajánlással együtt alkalmazandó.

1.3. Általános támadási technikák a MITRE ATT&CK Framework alapján

Az információbiztonsági támadások anatómiájának, illetve egy kiberbiztonsági támadás során alkalmazott támadási technikák megismeréséhez jó kiindulási pont a MITRE ATT&CK keretrendszer.
A MITRE ATT&CK keretrendszere leírja, miként kompromittálják a támadók az informatikai hálózatokat és rendszereket, és a rendszerekbe behatolva hogyan végzik tevékenységüket. Az ATT&CK a támadók szemszögéből vizsgálja a támadásokat, azt, hogy milyen célokat próbálnak elérni, és milyen konkrét módszereket alkalmaznak.

Ahogy az eredeti projekt, úgy a jelenlegi rendszer is azt a célt szolgálja, hogy jobban megértsük, hogy az ellenfelek hogyan készülnek fel a támadásokra és hajtják végre azokat, ami segít megtervezni a támadások kivédését vagy észlelését. Az ATT&CK keretrendszerben összegyűjtött tudás a védelmi képesség értékelésére, a gyenge pontok megtalálására és megerősítésére is használható.

A taktikák azt a remélt hatást jelölik, amiért a támadók a tevékenységüket végzik. A vállalati hálózatok és mobileszközök esetén hasonlóak a taktikák, a magasszintű célok, amit a támadók el akarnak érni.

A technika egy módszer egy „taktika” eléréshez, ami általában egyetlen lépés a tevékenységek sorozatában, amit a támadó küldetésének elérésének érdekében végez. Az ATT&CK sok egyéb mellett minden technikához tartalmaz leírást, példát, hivatkozást és javaslatot az adott támadási módszer megelőzésére és észlelésére.

1.4. Leggyakrabban alkalmazott általános támadási technikák

Jogosulatlan hozzáférés

Jogosulatlan hozzáférés szerzését sok tényező lehetővé teheti egy támadó részére. Például ilyen támadásra ad lehetőséget egy rosszul konfigurált webszerver, vagy egy gyenge kódminőségű weboldal (ami a hozzáférési jogosultságokat nem megfelelően kezeli, vagy nem végez valamilyen bevitel ellenőrzést megfelelően).

Ennek elkerülésére fontos, hogy a konfigurációk, a weboldalak és egyéb egyedileg fejlesztett kódok ellenőrizve legyenek nemcsak funkcionalitás, hanem biztonsági szempontok szerint is. A megfelelő szintű naplózás is elengedhetetlen akár egy biztonsági incidens detektálásához, akár annak visszafejtéséhez.

Kártékony kódok (Malware)

A kártékony kód (malware) egy általános megnevezés, amely minden kártékony, támadási céllal készített programkódot magában foglal. Ide tartoznak például a vírusok (virus), férgek (worm), kémprogramok (spyware), zsarolóprogramok (ransomware), agresszív reklámprogramok (adware).

A kártékony kódok legelterjedtebb fajtái napjainkban:

- ***Infostealer („információ lopó") programok***

  Ezek a programok mindenféle, az áldozat gépére telepített programokban tárolt jelszavakat próbálnak meg ellopni. Az infostealer programok célpontjai az e-mail kliensek, webböngészők, vagy akár adatátviteli kliensek (például FTP kliensek).

  Emellett az infostealer kódok egyéb funkciókat is megvalósíthatnak, mint például billentyűleütések rögzítése és képernyőmentések készítése, vagy akár bitcoin wallet lopás.

- ***Trójai programok (Trojan, RAT = Remote Access Trojan)***

  Általában legitim programoknak vannak álcázva, a fertőzés után alvó állapotban várakoznak a rendszeren a támadótól kapott utasítások érkezéséig.
  Funkciókat tekintve általában a következőkre képesek:
  - Botnet hálózathoz kapcsolás (például spammelés vagy Bitcoin-bányászat céljából)
  - Elektronikus pénzlopás
  - Adatlopás
  - Egyéb (káros) szoftver telepítése
  - Fájlműveletek (létrehozás, módosítás, törlés)
  - Képernyőfigyelés, képernyőmentés készítése
  - Webkamera figyelése, irányítása, felvétele és fénykép készítése
  - Távoli irányítás
  - Proxyszerver üzemeltetése
  - Banking trojans (Céljuk elsősorban a banki jelszavak és hitelkártyaadatok megszerzése)

- ***Zsarolóprogramok (ransomware)***

  Ezeknek a programoknak a célja az anyagi haszonszerzés „váltságdíj" formájában. Az áldozat számítógépén található fájlokat titkosítja, majd pénzt kér azok feloldásáért, általában lenyomozhatatlan módon, például kriptovalután keresztül.

- ***Makróvírusok***

  Ezek makró nyelven írt kártékony kódok, amelyek más programokba ültetve futnak le. A makróvírusok leggyakrabban MS Office dokumentumokba ágyazva érkeznek az áldozat eszközére. Gyakran komplex fertőzési láncolat elemeként alkalmazzák, például egy adathalász e-mailből megnyitásra kerül egy makróvírussal fertőzött MS Word dokumentum, amely a támadó által irányított távoli eszközről letölt például egy infostealer, vagy ransomware kódot.

Social Engineering

A social engineering (manipuláció) olyan támadás, amely során egy jogosultsággal rendelkező felhasználó jogosulatlan személy számára bizalmas adatokat ad át, vagy lehetőséget biztosít a rendszerbe történő belépésre a támadó megtévesztő viselkedése miatt. A social engineering leggyakoribb formái:

- ***Adathalászat (phishing)***

Ebben az esetben látszólag megbízható partnerként elektronikus levélben vagy honlapon próbálnak bizalmas információhoz jutni. A támadó által megszerezni kívánt információk a legkülönbözőbbek lehetnek például: felhasználónév, jelszó, hitelkártyaszám, bankszámla adatok stb. Az üzenet arra hívja fel a felhasználót, hogy jelentkezzen be valamilyen legitim honlaphoz (PayPal, eBay, valamilyen ismert bank stb.) nagyon hasonlító weboldalra és azon keresztül bizalmas információt adjon meg, ami valójában a támadó által üzemeltetett honlap.

- ***Telefonos adathalászat (vishing)***

  A telefonos adathalászat lényege, hogy a támadó valamilyen hangátviteli technológián (telefon, webes hanghívás, stb.) keresztül lép kapcsolatba az áldozattal. A hívó általában hitelkártya információt, esetleg más személyiség lopásra alkalmas információt igyekszik ezzel a módszerrel megszerezni. A telefonos adathalászat történhet úgy, hogy a felhasználót elektronikus levéllel keresik meg, és arra kérik, hogy hívja fel az ott megadott telefonszámot.

- ***Pharming***

  A pharming során a támadó a felhasználókat valódi webhelynek látszó hamis oldalakra irányítja, azáltal, hogy valamilyen módon átirányítja a hálózati forgalmát. Ennek eredményeként, amikor a felhasználó a valósnak vélt webcímet írja be, a rendszer egy hamis webhelyre irányítja át, amely hasonlít az igazira oldalra. Ezután a támadók vagy megpróbálnak hozzáférni személyes és pénzügyi adatokhoz, vagy megfertőzik az eszközt további kártékony kódokkal.

- ***Üzleti e-mail kompromittálása (BEC – business e-mail compromise):***

  Az ilyen típusú támadások során a támadók e-mailt küldenek, hogy elektronikus átutalásokat, vagy egyéb információk átadását sürgessék. Ezek a támadók gyakran hozzáférnek egy vezető e-mailjéhez, vagy hamisított e-mailen keresztül vezetőknek adják ki magukat. A támadók előzetesen felkutatják a lehetséges célpontokat és szervezeteiket, és mivel látszólag az általuk küldött e-mailek vezérigazgatóktól vagy más vezetőktől érkeznek, az alkalmazottakban nem merül fel gyanú a feladat végrehajtásával kapcsolatban.

Misuse („nem megfelelő használat”)

Olyan nem szándékos emberi tevékenységet értünk ez alatt, ami egy adott cég belső irányelveinek, illetve szabályzatának nem felel meg, például:

- jogosulatlan fájl másolás (üzleti titoknak minősülő adatok külső mobil adathordozóra másolása),
- nem jóváhagyott szoftver letöltése és telepítése,
- zene, vagy egyéb média jogosulatlan használata,
- P2P fájl megosztás (Torrent),
- távoli hozzáférést biztosító alkalmazások használata (például AnyDesk, Teamviewer),
- céges e-mail fiók használata magán célra,
- személyes célú web böngészés céges hálózatból.

2. Hacker támadások megelőzése

A hacker támadások megelőzésére számos folyamat és technológia szintű kontrollt alkalmaznak a szervezetek. Ebben a fejezetben ezek közül a legfontosabbak kerülnek bemutatásra.

Hozzáférés management

A hozzáférés-vezérlés olyan biztonsági mechanizmusok gyűjteménye, mely meghatározza, hogy a felhasználók mit tehetnek a rendszerben, azaz milyen erőforrásokhoz férhetnek hozzá és milyen műveleteket hajthatnak végre.

A hozzáférés vezérlés azért fontos, mert ha egy támadás során egy felhasználói fiókhoz hozzáfértek a támadók, akkor minimalizálható az általuk okozható kár.

A hozzáférés-vezérlés során alkalmazott fő elvek:

- Feladatok szétválasztása (Separation/Segregation of Duties)

  Célja, hogy egy folyamat lépéseit különböző személyek végezzék el. Ehhez a folyamatot meg kell tervezni, meg kell akadályozni, hogy egy személy a teljes folyamatot ellenőrizze és manipulálja.

- Legkevesebb jogosultság (Least Privilege)

  Az elv betartásával a rendszer a felhasználók és az alkalmazások erőforrásokhoz való hozzáférését csak a legszükségesebbekre korlátozza.

Hálózati szeparáció

A hálózatot úgy kell megtervezni, hogy az egyes eszközök csak azokkal az eszközökkel tudjanak kommunikálni, amik feltétlenül szükségesek ahhoz, hogy a funkciójukat ellássák. A szeparáció megvalósításának egyik technológiai módja a tűzfalak alkalmazása, míg a hálózati forgalomban kártékony tevékenység azonosítására és esetleg beavatkozásra az IDS és IPS rendszerek szolgálnak.

Tűzfal

A tűzfal olyan hálózati biztonsági rendszer, amely felügyeli és szabályozza a bejövő és kimenő hálózati forgalmat előre meghatározott biztonsági szabályok alapján. Jellemzően akadályt képez egy megbízható belső hálózat és egy nem megbízható külső hálózat között.

A forgalom szabályozásának részeként több funkciót is elláthatnak a tűzfalak, például csomagszűrést, állapot szerinti szűrést, alkalmazásszintű tűzfal funkciót, tartalomszűrést.

IDS

Az IDS (Intrusion Detection System) behatolás érzékelő rendszer. Az IDS legfontosabb célja és feladata, hogy azonosítsa a hálózatban a gyanús vagy kártékony tevékenységeket.

Két fajtája van:

- Host alapú IDS (HIDS), amely egy önálló rendszer tevékenységének a figyelésére szolgál. Ilyen például egy levelezőrendszer, egy webszerver vagy egy végfelhasználói számítógép. Csak a saját gazdájával foglalkozik nincs kapcsolata a környezetével.
- Hálózatalapú IDS (NIDS), amely egy számítógép-hálózat tevékenységét ellenőrzi. Általában a hálózati switcheken átmenő forgalmakat figyeli, monitorozza. Tipikus esetben nem tudja, hogy a rendszerhez tartozó egységeken belül mi történik.

IPS

Az IPS (Intrusion Prevention System) behatolásmegelőző rendszer. Működése nagyon hasonló az IDS-hez, a különbség az, hogy egy IDS a támadást csak felismeri, míg az IPS meg is próbálja blokkolni azt.

(D)DOS támadás elleni védelem

A szolgáltatásmegtagadással járó támadás (Denial of Service vagy DoS), más néven túlterheléses támadás, illetve az elosztott szolgáltatásmegtagadással járó támadás (Distributed Denial of Service, DDoS) informatikai szolgáltatás teljes vagy részleges megbénítása, helyes működési módjától való eltérítése.

A védekezés kis léptékben megoldható tűzfalakkal, viszont komoly sávszélességű támadás ellen igen költséges lehet, ezért arra készülve érdemes külső szolgáltatót választani.

Antivirus szoftverek

Az antivírus szoftverek célja a kártékony kódok (malware) felismerése és azok károkozásának ellehetetlenítése.

Alapvetően kétféle megoldás létezik belőle:

- Mintaillesztéses (Signature based): Ezek régebbi technológiát képviselnek, a kártékony kódokat azok szignatúrája alapján azonosítja. Előnyük, hogy kevésbé erőforrásigényesek. Hátrányuk, hogy viszonylag könnyű egy adott kártékony kódot úgy módosítani, hogy azt ne ismerjék fel.
- Viselkedés alapú (Behaviour based): Ez a modell már egy új, fejlettebb detekciós mechanizmus. A vírusok közös, gyanús tulajdonságai alapján próbálják meg azokat felismerni és blokkolni. Például bizonyos sorrendű API hívások alapján, vagy registry bejegyzés létrehozása/fájlrendszeren létrejövő fájl alapján stb. Előnye, hogy több gyanús mintát észlel kevesebb hamis riasztás mellett.

Biztonsági incidens és esemény kezelő rendszerek

A biztonsági incidensek és események kezelése (SIEM - Security Incident and Event Management) alatt a biztonsági incidensek vagy események azonosítását, megfigyelését, rögzítését és elemzését értjük, valós idejű IT környezetben.

A SIEM a felügyeleti és detekciós funkciót a biztonság szempontjából releváns naplóbejegyzések, hálózati forgalom és felhasználói fiókok tevékenységének és viselkedésének elemzésén keresztül valósítja meg.

Hardening

A rendszersérülékenység csökkentésére alkalmazott eljárásokat nevezik hardening-nek. A támadási felület redukálására az alábbi módszereket szokták használni általánosságban:

- alapértelmezett jelszavak megváltoztatása,
- a rendszer funkcionalitásához nem szükséges szoftverek eltávolítása,
- felesleges fiókok eltávolítása,
- felesleges szolgáltatások kikapcsolása vagy eltávolítása, feleslegesen nyitott portok lezárása.

Az alábbi helyeken lehet hardening guide-okat találni:

- NIST:
  - https://nvd.nist.gov/ncp/repository
  - https://csrc.nist.gov/publications/detail/sp/800-123/final
- CIS:
  - http://benchmarks.cisecurity.org

Konfiguráció menedzsment

A konfiguráció menedzsment (configuration management) célja rendszeresen auditálni a szolgáltatások beállításait a hibás konfiguráció feltárása céljából.

Rendszerek elemeinek frissítése

Az egyik legfontosabb biztonsági üzemeltetési feladat az elektronikus információs rendszerek sérülékenységeinek javítása biztonsági frissítésekkel.

Penetration testing

Informatikai rendszer egy vagy több elemének olyan ellenőrzése (részben manuális úton), melynek során megállapítható, hogy van-e a gyakorlatban kihasználható, ismert sérülékenysége.

Adatszivárgás-megelőzés (Data Leak Prevention - DLP**)**

A DLP rendszereket arra tervezték, hogy észleljék és megakadályozzák a bizalmas információk jogosulatlan használatát, továbbítását, mind a végpontok, a hálózat és az adattárolók tekintetében.

Biztonsági mentés (Backup)

Talán mind közül a legfontosabb, hogy a kritikus adatokról készüljön biztonsági mentés, különösen olyan, ami hálózatról semmilyen formában sem érhető el (cold backup). Többek között ez zsarolóvírusok esetében tud hasznos lenni, ha már megtörtént a baj.

3. Sérülékenységek

A sérülékenységkezelés célja a rendszereket érintő sérülékenységek azonosítása és az azok által hordozott kockázatok csökkentése a sérülékenység megszűntetésével, vagy kompenzációs kontroll alkalmazásával.

3.1. Sérülékenység elemzés

Az új nyilvánosságra kerülő sérülékenységekről számos forrásból lehet értesülni, például a CVE, vagy NVD adatbázisokból. A CVE kategorizálva, egyedi azonosítószámmal ellátva ad alapvető információkat az egyes sérülékenységekről. A CVE-ből érkező információkból épül az NVD adatbázis, ami már kibővíti az adott sérülékenységhez tartozó adatokat az ismert javítási lehetőségekkel, valamint súlyosságra és lehetséges hatásokra vonatkozó információkkal.

3.2. Sérülékenység vizsgálat

A sérülékenység vizsgálat egy technikai módja a szervezetet érintő sérülékenységek feltárásának, ezért ez a sérülékenység elemzés kiegészítésének tekinthető. Egy nagyvállalati környezetben a sérülékenység vizsgálat számos nehézségbe ütközhet, többek között:

- mielőtt megtörténne maga a vizsgálat, megfontolt tervezésre van szükséges a vizsgálat megfelelő idejének meghatározására, hogy az esetleges üzleti hatásokat minimalizálni lehessen, mert akár még passzív vizsgálatok is negatívan befolyásolhatják a rendszerek működését,
- egy nagyvállalati környezetben egy vizsgálat teljes lefutása hosszú időt (heteket, vagy akár hónapokat) is igénybe vehet az infrastruktúra nagy mérete miatt. Ha a sérülékenység

vizsgálatot nem egészíti ki sérülékenység elemzés is, akkor két vizsgálati ciklus között számos sérülékenységről hosszú ideig nem szerez tudomást a szervezet,
- néhány kritikus, vagy egyedi technológiákat használó környezetben a tervezés és megvalósítás további körültekintést kíván.

3.3. Sérülékenységek priorizálása és osztályozása

Ha egy sérülékenységről megállapításra kerül, hogy érinti a szervezet valamely információs eszközét, meg kell határozni, hogy milyen kockázatokat képvisel, hogy ezek alapján a válaszlépéseket, vagy a kompenzációs kontrollok megvalósítását priorizálni lehessen.

A sérülékenységek osztályozásának (vulnerability triage) célja a válaszlépések elvégzéséhez rendelt határidő meghatározása.

A priorizálás nem alapulhat kizárólag a sérülékenység súlyosságán, figyelembe kell venni az érintett eszköz szervezeten belüli kritikusságát is. Ezért a sérülékenység priorizálásának módja, amit a szervezet alkalmaz, mindkettőt figyelembe kell vegye.

Az úgynevezett 0-day sérülékenységek, amelyekre még nem létezik javítás, de létezik már az adott sérülékenységet kihasználó kártékony kód (exploit), kiemelt körültekintést igényelnek a priorizálás során, mert néha túlzott figyelmet kapnak valós hatásuk és kockázati kitettségükhöz képest.

3.4. Sérülékenységek hatásának csökkentése

Ha egy információs eszközről megerősítésre kerül, hogy sérülékeny, és az adott sérülékenység ki is használható, akkor meg kell határozni, milyen módon csökkenthető az adott sérülékenység hatása.

Ha rendelkezésre áll biztonsági frissítés, akkor a megállapított prioritással összhangban el kell végezni a patch kezelését. Ellenkező esetben valamilyen kompenzációs kontroll definiálásra van szükség. Ha létezik a sérülékenység hatásának csökkentéséről nyilvánosan elérhető információ (például a sérülékeny rendszer gyártójától), ez alapul szolgálhat, de minden esetben ellenőrizni kell annak a szervezeten belüli alkalmazhatóságát.

Még abban az esetben is szükség lehet kompenzációs kontroll definiálására, ha van rendelkezésre álló biztonsági frissítés a sérülékenységhez. Nagyvállalati környezetben ugyanis a frissítések kezelése (patch management) számos nehézségbe ütközhet, mint például:

- milyen módon legyen priorizálva az adott biztonsági frissítés az egyéb, még nem alkalmazott javítócsomagokhoz képest;
- mekkora erőforrásigénye van a biztonsági frissítés ellenőrzésének;
- milyen módon található jóváhagyott időablak a biztonsági frissítés alkalmazásához;
- hogyan kezelhetőek a változatos környezetek, hordozható eszközök, virtualizált környezetek, nem sztenderd IT komponenseket tartalmazó környezetek (például appliance eszközök, ahol a komponensenkénti frissítés a gyártó nem támogatja).

4. Kódolások biztonsági kérdései

Az elmúlt néhány évben a szoftverekre leselkedő fenyegetések exponenciálisan növekedtek. Szoftverek és applikációk sebezhetőségei sok kárt okoztak vállalatoknak és embereknek.

A biztonságos kódolási normák és legjobb gyakorlatok lehetővé teszik a fejlesztők számára, hogy biztonságosabb szoftvereket készítsenek. Ezek a normák biztosítják, hogy a fejlesztők oly módon írják meg alkalmazásaik kódjait, hogy azok támadók által kihasználható sebezhetőségektől mentesek legyenek.

Bár biztonságos alkalmazások fejlesztésének számos módja létezik, az OWASP (Open Web Application Security Project) létrehozott egy átfogó, biztonságos kódolást biztosító ellenőrző listát. Ez a lista főleg webalkalmazásokra fókuszál, de alkalmazható biztonsági protokollként minden szoftverfejlesztési életciklusban vagy szoftvertelepítési platformban a rossz kódolási gyakorlatokból eredő veszélyek minimalizálására.

Az OWASP a következő biztonságos kódoláshoz szükséges listát állította össze, mely tartalmaz számos prevenciós technikát a különböző fajtájú támadásokból keletkező kár minimalizálására és kezelésére.

Input ellenőrzés

A felhasználó vagy az alkalmazás által bevitt inputok, adatok ellenőrzése.

Az ellenőrzéssel kiszűrhetőek olyan paraméterek az inputokból, amelyek abnormális viselkedést váltanának ki az alkalmazásból. Néhány módszer az input ellenőrzésére:

- Az ellenőrzésnek megbízható rendszeren kell történnie.
- Ellenőrzés adattípus, lehetséges érték és hossz alapján, valamint a beérkező input összehasonlítása a megengedett karakterekkel.
- Kérések fejléceinek vizsgálata és ellenőrzése, hogy azok csak ASCII karaktereket tartalmaznak.
- Minden adat ellenőrzése, ami klienstől/felhasználótól érkezik, beleértve az URL-eket, HTTP fejléceket, beágyazott kódokat stb.

Output kódolás

Output kódolás során a felhasználói inputot/nem biztonságos adatokat alakítunk át olyan formára, hogy az input megjelenítése során az biztosan ne kerüljön futtatásra a böngésző vagy program által. Az output kódolás használata kizárja, hogy megbízhatatlan forrásból érkező adat, melyet az interpreter nem ért meg, futtatásra kerüljön.

A cross-site scripting kivédhető output kódolás használatával.

Néhány technika az OWASP ajánlásaiból:

- Az ellenőrzésnek megbízható rendszeren kell történnie.
- Minden karakter kódolása, kivéve, ha azok vélhetően biztonságosak az interpreter számára.
- Nem megbízható adatok megtisztítása OS parancsokkal.

Hitelesítés és jelszómenedzsment

Hitelesítés során ismerjük meg a felhasználó kilétét. A jelszómenedzsment szabályok és technikák sorozatának betartatását jelenti jelszavak tárolása során. Ezen szabályok követése által férhetnek hozzá a felhasználók bizalmasan bizonyos kritikus eszközökhöz. OWASP által ajánlott technikák erre vonatkozóan:

- Hitelesítés igénylése minden eszközhöz, weboldalhoz, amelynél számít a bizalmasság, integritás, elérhetőség.
- Tartózkodás a jelszavak újra használatától.
- Felhasználók értesítése jelszó visszaállításokról.

- Hibás bejelentkezési kísérletek számának jelzése a felhasználó felé a következő sikeres belépéskor.
- Rövid lejárati idő az ideiglenes jelszavaknak, melyek megváltoztatása kötelező a következő belépéskor.
- Jelszókomplexitás előírása és betartatása

Session Management

Szolgáltatásokban vagy webalkalmazásokban több különböző felhasználó által megadott kérés biztonságos kezelését nevezzük session managementnek. Erre vonatkozó tanácsok az OWASP-tól:

- Session-ök és kapcsolatok teljes megszüntetése kilépéskor.
- Egy felhasználói azonosítóval csak egy bejelentkezés engedélyezése.
- A kockázatok és az üzleti célok figyelembevételével a munkamenet inaktivitási időkorlátjának a lehető legrövidebbnek kell lennie.

Kriptográfiai gyakorlatok

A kriptográfiai műveleteket általában a bizalmasság megőrzése érdekében alkalmazzák, hogy az érzékeny adatokat csak azok a felhasználók láthassák és módosíthassák, akiknek szükséges. Kriptográfiai gyakorlatokra vonatkozó tanácsok:

- Kriptográfiai műveletek alkalmazása egy megbízható rendszeren, a szenzitív adatok bizalmasságának érdekében.
- A véletlenszámok, fájlnevek, GUID-ok (Global Unique Identifier – globális egyedi azonosító) generálásához elfogadott, jóváhagyott random szám generátor használata.
- A kriptográfiai kulcskezelést irányelvek és folyamatok kidolgozásával és követésével kell alkalmazni.
- Mesterkulcsok védelme a jogosulatlan hozzáférésektől.

Hibakezelés és naplózás

Hibakezelés során olyan esetekkel kapcsolatos eljárásokról beszélünk, melyek során nemvárt kimenetet ad az alkalmazás/rendszer, ezek általában abnormális input eredményei. A naplózás lehetővé teszi, hogy a szoftver vagy alkalmazás változásait nyomon kövessük. Ezekre vonatkozó OWASP tanácsok:

- Hibakezelés során ne adjunk vissza információt a rendszer működésére vonatkozó információt.
- Hibák előfordulása esetén, a memóriát megfelelően fel kell szabadítani.
- A naplók ne tároljanak a rendszerre vonatkozó szenzitív információt.
- Meghiúsult input ellenőrzéssel, hitelesítési próbálkozásokkal, hozzáférés- és kivételkezeléssel és biztonsági konfiguráció változtatásokkal kapcsolatos naplók karbantartása és ellenőrzése szükséges.

Adatvédelem

Adatvédelem az a folyamat, amely megóvja az adatokat azok megváltoztatásától, kompromittálódásától, vagy az elvesztésétől. A bizalmasság, integritás és elérhetőség megőrzése mellett az adatok a következő technikákkal védhetők:

- Kövessük a legkisebb jogosultság (least privilege) alapelvét. Limitáljuk a felhasználók jogosultságát és hozzáférését az adatokhoz, rendszerekhez és funkciókhoz úgy, hogy csak azokat érjék el, amelyek a feladataik elvégzéséhez nélkülözhetetlen.

- Bizalmas adatok kizárása a HTTP GET kérésekből.
- Védjük a szerveroldali kódokat és előzzük meg, hogy sima felhasználók számára elérhetők legyenek. Kritikus és érzékeny adatok hozzáférését szabályozni kell.
- Alkalmazások vagy weboldalak űrlapjaiba történő adatbevitel során az autocomplete funkció kikapcsolása.

Kommunikáció biztonság

Minden szenzitív információt titkosítással kell védeni. Kapcsolatok védelme érdekében a TLS (Transport Layer Security) használható más titkosítási algoritmusokkal közösen.

- TLS használata esetén a sikertelen kapcsolatok ne váltsanak kevésbé biztonságos protokollra.
- TLS használata külső forrásokból származó érzékeny adatok védelmére.

Rendszerkonfiguráció

A következő lista követését ajánlja az OWASP rendszerek konfigurációjakor:

- Gondoskodjunk arról, hogy a rendszerek, keretrendszerek és rendszerkomponensek legfrissebb verziói fussanak, amelyek tartalmazzák a javításokat.
- A legkisebb jogosultság elvének alkalmazása webszerverekre, service accountokra és folyamatokra.
- A HTTP válaszok fejlécei csak a legszükségesebb információkat tartalmazzák. OS, webszerver verzió és szoftver keretrendszer információk ne jelenjenek meg.
- A teszt és fejlesztői környezetek legyenek izolálva az éles környezettől.

Adatbázis biztonság

Adatbázisokkal kapcsolatos veszélyek megelőzésére tett tanácsok:

- Adatbázis hozzáférések során az alkalmazás használja a lehető legalacsonyabb jogosultsági szintet.
- Alapértelmezett jelszavak azonnali megváltoztatása.
- Többfaktoros azonosítás, ahol lehetséges.
- A nem feltétlenül szükséges alapértelmezett felhasználók tiltása.

File menedzsment

Az OWASP által ajánlott lépések:

- Hitelesítést minden szerverre történő fájlfeltöltés esetén.
- Szerverre feltöltött fájlok ellenőrzése a fejlécük vizsgálatával.
- Futtatási jogosultság kikapcsolása azokban a mappákban, ahova fájlokat lehet feltölteni.
- Soha ne küldjük el az abszolút elérési utat a felhasználónak.

Memória menedzsment

A memória menedzsment a védelem különösen fontos aspektusa, mert sok támadás a memóriához kapcsolódik. A következő lépések betartása szükséges a megfelelő memória menedzsmenthez:

- Kerüljük a sebezhető függvények használatát (pl. print, strcat, strcpy stb.).
- Vizsgáljuk a puffer méretét túlcsordulásokkal szemben.

- Az input string-ek megfelelő levágása (truncate) szükséges mielőtt copy vagy concatenation függvényeket használnánk.

5. OWASP sérülékenységek

5.1. OWASP és az OWASP Top 10

Az Open Web Application Security Project (OWASP) egy nyílt közösség, mely eltökélt, hogy elősegítse megbízható alkalmazások és API-k fejlesztését, beszerzését és karbantartását.

OWASP Top10

Az OWASP Top 10 a webalkalmazások biztonsági szintjének emelésére fókuszál. Több alkalmazásbiztonsággal foglalkozó cég és szakértő közreműködésével azonosították a 10 legkritikusabb, webalkalmazásokat fenyegető kockázatot. A lista összeállítása közben figyelembe vették, hogy mennyire elterjedt a fenyegetés, mennyire használható ki a sérülékenység és mennyire bonyolult a detektálása, továbbá a sérülékenység kihasználásának, azaz egy sikeres kibertámadásnak, milyen hatásai lennének a cég pénzügyeire, rendszereire és hírnevére.

Az OWASP Top 10 listája jó alapot jelent a fejlesztőknek, hogy megismerjék a legfontosabb biztonsági problémákat, amelyeket el kell, és remélhetőleg el akarnak kerülni. Magasabb szinten a lista segít a felfedezett sérülékenységek rangsorolásában, így a vállalatok útmutatóként használhatják, amely megmutatja, hogy az erőforrásaikat melyik problémák megoldására kell összpontosítaniuk.

5.2. OWASP gyakorlati alkalmazása

Az OWASP Top 10 elsődlegesen egy tudatosságot erősítő dokumentum, ettől függetlenül sok szervezet, de facto alkalmazásbiztonsági sztenderdként használja. Nem szabad azonban elfelejteni, hogy az OWASP top 10 kódolási vagy tesztelési standardként a minimum, csak egy kiindulópont.

Mi az ASVS (Application Security Verification Standard - Alkalmazásbiztonsági ellenőrzési sztenderd)?

Az OWASP azt tanácsolja, hogy bárki, aki alkalmazásbiztonsági szabványt szeretne alkalmazni, használja az általuk kidolgozott Application Security Verification Standard (ASVS)-t, amit úgy terveztek, hogy ellenőrizhető és tesztelhető legyen és a biztonságos alkalmazásfejlesztési életciklus bármelyik részében alkalmazható legyen.

Az ASVS alkalmazásbiztonsági követelmények és tesztek gyűjteménye tervezők, fejlesztők, tesztelők, biztonsági szakértők, eszközgyártók és ügyfelek számára biztonságos alkalmazások tervezéséhez, fejlesztéséhez, teszteléséhez és ellenőrzéséhez.

6. NIST iránymutatásai fejlesztők részére

6.1. Szervezet felkészítés

A szervezetnek biztosítania kell, hogy az emberei, a folyamatai és a technológiája elég felkészült legyen a biztonságos szoftverfejlesztéshez.

Rendszerfejlesztés biztonsági követelményeinek meghatározása

Gondoskodni kell arról, hogy a szoftverfejlesztés biztonsági követelményei mindig ismertek legyenek a szervezet számára, tehát első lépésként ezeket a követelményeket össze kell gyűjteni és megosztani az illetékesekkel. A belső (pl.: a szervezet irányelvei, üzleti céljai és kockázatkezelési stratégiája) és külső forrásokból (pl.: vonatkozó törvények és rendeletek) származó követelményeket is figyelembe kell venni.

1. Meg kell határozni és dokumentálni a szervezet szoftverfejlesztési infrastruktúráira és folyamataira vonatkozó összes biztonsági követelményt.
2. Meg kell határozni és dokumentálni kell a szervezet által fejlesztett szoftverekkel szemben támasztott összes biztonsági követelményt, hogy megfeleljenek az elvárásoknak.
3. A követelményeket közölni kell minden olyan harmadik féllel, aki szoftver komponenseket biztosít a szervezetnek, amit a szoftverekben felhasználnak.

Szerepkörök és felelősségek meghatározása

Gondoskodni kell arról, hogy mindenki, aki részt vesz a szoftverfejlesztés életciklusában, tisztában legyen a saját szerepkörével és az ehhez tartozó kötelezettségeivel, valamint a biztonsági követelményekkel.

1. A szerepköröket és a hozzájuk tartozó kötelezettségeket és felelősségeket úgy kell meghatározni, hogy a szerepkörök lefedjék a teljes szoftverfejlesztési életciklust. Ezeket a szerepköröket rendszeresen felül kell vizsgálni és szükség szerint frissíteni kell a feladataikat és kötelezettségeiket.
2. Az egyes szerepkörök felelős személyei számára képzéseket kell biztosítani a biztonságos fejlesztés érdekében.
3. A felsővezetést elkötelezetté kell tenni a biztonságos fejlesztés mellet, és ezt az elkötelezettséget közvetíteniük kell a biztonságos szoftverfejlesztési folyamat minden szereplője felé.

Szoftver biztonsági ellenőrzéseinek meghatározása és használata

A szoftverfejlesztés életciklusa során keletkezett terméknek meg kell felelnie a szervezet elvárásainak, tehát feltételeket kell meghatározni és alkalmazni a szoftver biztonságának ellenőrzése érdekében.

1. A szoftverek biztonsági ellenőrzéséhez olyan feltételeket kell meghatározni, amelyek lefedik a teljes fejlesztési életciklust.
2. Használjunk olyan folyamatokat és megoldásokat, melyek összegyűjtik és tárolják a szükséges információkat arról, hogy a szoftver a feltételeknek megfelel.

Biztonságos környezet megvalósítása és karbantartása a fejlesztéshez

Biztosítani kell, hogy a szoftverfejlesztés során minden folyamat (pl.: a fejlesztés, tesztelés, üzemeltetés) védett legyen a belső és külső fenyegetésekkel szemben.

1. El kell különíteni, és meg kell védeni a szoftverfejlesztésben részt vevő különböző környezeteket.
2. A fejlesztési végpontok tevékenységeit (pl.: tervezők, fejlesztők, tesztelők) a fejlesztéssel kapcsolatos feladataik végrehajtásához biztonságossá kell tenni.

6.2. Szoftver védelme

A szervezetnek meg kell védenie a szoftver minden komponensét a kód manipulációjától és az illetéktelen hozzáféréstől.

A kód védelme a jogosulatlan hozzáféréstől és manipulációtól

Meg kell előzni a kód gondatlan, vagy szándékos jogtalan módosításait, amellyel megkerülhetők a szoftver tervezett biztonsági előírásai.

A nyilvánosan nem hozzáférhető kód segít megelőzni a szoftver eltulajdonítását és megnehezíti, időigényesebbé teszi a támadók számára az esetleges sebezhetőségek felfedezését és kihasználását.

A kódot a legkisebb jogosultság elve alapján tároljuk, csak az arra jogosult személy férhessen hozzá a számára szükséges részlethez.

Mechanizmus biztosítása a szoftverkiadás integritásának ellenőrzésére

A szoftvert megvásárlóknak és használóknak lehetőséget kell adni, hogy meggyőződhessenek arról, hogy a termék legitim és nem manipulált, a hitelesítő információkat számukra elérhetővé kell tenni.

Minden szoftververzió archiválása és védelme

A különböző szoftver kiadásokat (verziókat) biztonságosan el kell tárolni (archiválni), hogy az azokban felfedezett sebezhetőségeket a későbbiek során detektálni, analizálni és javítani lehessen. Az egyes verziókhoz tartozó hitelesítő információkat és származási adatokat is archiváljuk a szükséges fájlokkal és egyéb adatokkal együtt.

6.3. Biztonságos szoftverfejlesztés

A szervezetnek jól védett szoftvereket kell előállítania, amelyek a lehető legkevesebb sérülékenységet és biztonsági problémát tartalmaznak.

A biztonsági követelményeknek megfelelő szoftver tervezése és a biztonsági kockázatok csökkentése

A szoftver biztonsági kockázatainak felméréséhez különböző kockázatmodellezési megoldások használhatók, például fenyegetés modellezés vagy támadás modellezés. A szoftverek biztonsági követelményeit, a felmerülő kockázatokat és a tervezés ehhez kapcsolódó döntéseit dokumentálni kell.

A szoftvertervezet áttekintése a biztonsági követelményeknek és kockázati információknak való megfelelés ellenőrzéséhez

Biztosítani kell, hogy a szoftver megfeleljen a felállított biztonsági követelményeknek és megfelelően kezelje az azonosított kockázati tényezőket. Olyan szakképzett személyt bevonva, aki nem vett részt a tervezésben, megfelelő automatizált folyamatokkal meg kell vizsgáltatni, hogy a szoftver megfelel-e az összes biztonsági követelménynek.

Meglévő, jól védett szoftverek újra felhasználása, a funkcionalitás megkettőzése helyett

A szoftver egyes moduljainak és szolgáltatásainak (amelyek biztonságosságát már ellenőrizték) újra felhasználásával csökkenteni lehet a szoftverfejlesztés költségeit és fel lehet gyorsítani a fejlesztés folyamatát. Amennyiben egyetlen külső forrásból származó eszköz sem tud megfelelni a fejlesztési és biztonsági igényeiknek, úgy saját fejlesztésű eszközt kell létrehozni, aminek szintén meg kell felelnie az előírt biztonsági és működési követelményeknek.

Az összes belső és külső forrásból származó komponens eredetét és integritását ellenőrizni kell, mielőtt újra felhasználnánk.

Forráskód létrehozása a biztonságos kódolási gyakorlatok betartásával

A szervezet által meghatározott kritériumokat végig szem előtt tartva kell elkészíteni a forráskódot. Követni kell a fejlesztés során alkalmazott programozási nyelvek, fejlesztői környezetek biztonságos kódolási gyakorlatát (pl.: inputok ellenőrzése, outputok kódolása, nem biztonságos függvények és függvényhívások használatának mellőzése, megfelelő hibakezelés, naplózás, változáskövetés stb.).

Az integrált fejlesztői környezet, a fordítási folyamatok konfigurálása a biztonság javítása érdekében

Olyan forráskód fordító-, értelmező-, összeállító eszközöket kell alkalmazni, amelyek a keletkező futtatható szoftver biztonságát javító funkciókat kínálnak. Meg kell határozni, hogy ezen eszközök mely funkcióit kell használni, az eszközöket ennek megfelelően kell konfigurálni (pl.: fordítás során engedélyezzük a fordítóprogramnak, hogy üzenetet küldjön, ha rosszul védett kódot talál).

Ember által olvasható kód értelmezése és elemzése a sebezhetőségek azonosítása és a biztonsági követelményeknek való megfelelés ellenőrzése érdekében

Az esetleges sebezhetőségeket kódelemzéssel, kódellenőrzéssel azonosítani kell, hogy ezeket még a szoftver kiadása előtt ki lehessen javítani. Ez a folyamat automatizált eljárásokkal hatékonyabbá tehető.

Futtatható kód tesztelése a biztonsági rések azonosításához és a biztonsági követelményeknek való megfelelés ellenőrzéséhez

Tesztelés során is törekedni kell arra, hogy a sérülékenységeket és biztonsági réseket a szoftver minden komponensében detektálni, elemezni és dokumentálni lehessen, hogy a fejlesztőcsapat a szoftver kiadása előtt ki tudja javítani ezeket. A tesztelést meg kell tervezni, majd a tényleges tesztelés során minden bemenetet, eseményt, eredményt és problémát dokumentálni kell.

Szoftver alapértelmezett biztonsági beállításainak konfigurálása

A szoftver telepítésekor is szem előtt kell tartani a biztonsági követelményeket, hogy minimalizálni lehessen annak a valószínűségét, hogy a termék hibás biztonsági beállítások mellett kerüljön üzembe helyezésre. Ezeket a beállításokat dokumentálni kell és a dokumentáció el kell juttatni a rendszerüzemeltetőkhez.

7. Google Analytics kódok implementálásának feltételei IT biztonsági szempontból

7.1. Harmadik féltől származó frontend komponensek patch kezelése

Harmadik féltől származó komponensek kockázatai

A web applikációk fejlesztése során, mivel ezek nyilvánosan elérhetőek, különös figyelmet kell fordítani a biztonságra. A fejlesztőknek gondoskodni kell arról, hogy ezek az applikációk megfelelően védettek legyenek a fontosabb ismert biztonsági kockázatoktól.

Biztonságos webapplikációk fejlesztésrére irányuló erőfeszítéseket könnyen semmissé tehetik az olyan sérülékenységek, amelyek harmadik féltől származó komponensekben találhatók, például forráskódban scriptekben vagy CSS fájlokban.

A kockázatcsökkentés legjobb gyakorlatai

A harmadik féltől származó komponensek felhasználásához köthető kockázatok minimalizálása érdekében az alábbi lépéseket célszerű beilleszteni a fejlesztési folyamatba.

- Komponensek leltára
- Függőségek vizsgálata
- Kockázat értékelés
- Kockázat csökkentés

Komponensek leltára

Az első lépés során meg kell ismerni, hogy az applikáció milyen függő komponenseket használ. Ez alapvető fontosságú a projekthez köthető biztonsági kockázatok csökkentése érdekében. A folyamat segíti annak végig gondolását is, hogy valóban szükség van-e az egyes harmadik féltől származó komponensek használatára.

Függőségek vizsgálata

Miután elkészült a függő komponensek leltára a következő lépés a sérülékenységek keresése. Amennyiben a komponens nyílt forráskóddal rendelkezik, a kód könnyen vizsgálható. Amennyiben nem nyílt forráskódú a komponens, vizsgálhatóak az gyártó biztonsági jelentései, vagy kutatható speciális adatbázisokban. Természetesen nem célszerű az összes függő komponenst manuálisan vizsgálni, érdemesebb erre a célra fejlesztett automatizált eszközöket használni. Módszertől és eszköztől függetlenül a vizsgálat eredményének tartalmaznia kell a függő komponensek sérülékenységeinek listáját és ezek súlyossági fokát.

Kockázat csökkentés

Ha a sérülékenységre van megoldás, és a frissítés nem okoz problémát, akkor alkalmazni kell. Amennyiben az alkalmazás harmadik féltől származó függő komponensének sérülékenységére nincs elérhető megoldás, két megoldás lehetséges:

- Együttműködés a kód írójával/gyártóval a sérülékenyég javítása érdekében.
- A sérülékenységet megkerülő megoldás keresése.

A kockázat csökkentés kontroll alatt tartása

Az elfogadható kockázati szint elérése nem jelenti azt, hogy a probléma nem kerülhet elő ismét. Ha a saját kód nem is változik, a harmadik féltől származó komponensek bármikor változhatnak, és egy egyszerű újra buildeléssel is kerülhet ismeretlen kód az alkalmazásba.

Kliens oldali komponensek lezárása - külső erőforrás integritásának ellenőrzése

A sub-resource integrity egy biztonsági funkció, ami a böngésző számára lehetővé teszi annak ellenőrzést, hogy az általa kezelt komponensek (amelyek például egy Content Delivery Network-ről származnak) módosítás nélkül érkeznek-e meg. Működésének alapja egy kriptográfia hash érték, amellyel az általa kezelt komponensnek egyeznie kell.

Content Delivery Network-ök (a továbbiakban: CDN) használata olyan scriptek és stílus fájlok esetében, amelyek meg vannak osztva több oldal között is, hatékonyabbá tudja tenni az oldal működését és sávszélességet tud megtakarítani. Ugyanakkor a CDN-ek használata azzal a veszéllyel jár, hogy amennyiben egy támadó átveszi az irányítást egy CDN-ön, képessé válhat rosszindulatú kód hozzáadására, vagy akár az egész kód átírására. Ezáltal a támadó képes lehet támadni az összes olyan oldalt, amely au adott CDN-ről származó fájlokat használja.

Külső erőforrás integritásának ellenőrzésével csökkenthető az ilyen támadások kockázata, mivel biztosítja azt, hogy a webalkalmazás vagy weboldal külső forrásból származó fájljai úgy érkezzenek meg, hogy nem történt rajtuk módosítás és harmadik fél semmilyen tartalmat nem adott hozzájuk.

8. A tananyag elsajátítására vonatkozó nyilatkozat

A jelen tananyag elsajátítására vonatkozó nyilatkozatot az alábbi szöveggel kérjük a dora_oktatas@otpbank.hu e-mail címre eljuttatni:

"Az OTP Bank Nyrt. által az IKT szolgáltatónak minősülő Szerződő Partnerek számára a DORA rendelet előírásainak megfelelően kialakított oktatási anyagot a [Partner cég neve] által az OTP Bank Nyrt-nek nyújtott szolgáltatásban résztvevő valamennyi személy elolvasta, az abban foglaltakat megértette és magára nézve kötelezőként ismeri el. A [Partner cég neve] vállalja, hogy a szolgáltatás teljes életciklusa alatt biztosítja a szolgáltatás nyújtásába bevont munkavállalói és alvállalkozói részére az oktatási anyag megismertetését és elvárja munkavállalóitól és alvállalkozóitól az azokban foglaltak betartását."

9. Tananyag elsajátítására vonatkozó nyilatkozat egyéni vállalkozók részére

Amennyiben Ön egyéni vállalkozó, kérjük, hogy az alábbi tartalmú nyilatkozatot szíveskedjék kitölteni, és a dora_oktatas@otpbank.hu e-mail címre elküldeni: „A nyilatkozat kitöltésével kijelentem, hogy az OTP Nyrt. Általános Adatkezelési tájékoztatójában (https://www.otpbank.hu/portal/hu/adatvedelem) foglaltakat megismertem és annak tartalmát elfogadom."

Tájékoztatjuk továbbá, hogy az OTP Bank, mint adatkezelő a nyilatkozattétellel összefüggésben a DORA rendelet szerinti elvárásoknak való megfelelés érdekében az Általános Adatvédelmi Rendelet 6. cikk (1) bekezdés f) pontja szerinti jogos érdek jogalapján kezeli az egyéni vállalkozó nevét, e-mail címét, továbbá az oktatás elvégzésének tényét.