Odbor riadenia kybernetickej a informačnej bezpečnosti
Verzia 1.1 | kyberbezpecnost@mirri.gov.sk | jún 2023

MINISTERSTVO
INVESTÍCIÍ, REGIONÁLNEHO ROZVOJA
A INFORMATIZÁCIE
SLOVENSKEJ REPUBLIKY

Metodika pre verejnú správu pre bezpečný vývoj nových aplikácií a systémov v súlade so štandardom SSDLC

Obsah

1 Správa dokumentu

Tento dokument je metodickým materiálom slúžiacim pre potreby orgánov verejnej moci, ktorý nie je povinný na použitie a ani nie je záväzný. Dokument je poskytnutý voľne a bezplatne na využitie podľa potrieb konkrétnej organizácie.

Vytvorený dokument je možné použiť i pre potreby vzdelávania pracovníkov organizácií v oblasti kybernetickej a informačnej bezpečnosti.

Vytvorený dokument nie je určený na ďalší predaj alebo akúkoľvek inú komerčnú či obchodnú činnosť.

Ministerstvo investícií, regionálneho rozvoja a informatizácie Slovenskej republiky (ďalej aj „MIRRI“) nezodpovedá za nesprávne použitie predmetného dokumentu zo strany organizácie. Správne použitie a implementácia bezpečnostných opatrení je plne v kompetencii a zodpovednosti konkrétnej organizácie.

MIRRI si vyhradzuje právo na zmenu/úpravu predmetného dokumentu alebo čiastkových textov a tabuliek, a to v potrebnom rozsahu vrátane zmien verzií dokumentov. Dokument je výstupom pilotného projektu, na ktorý nadväzuje Reforma Štandardizácia technických a procesných riešení kybernetickej a informačnej bezpečnosti (Plán obnovy a odolnosti).

2 Úvod

2.1 Účel dokumentu

Je veľmi dôležité, aby bol softvérový produkt chránený pred akýmkoľvek druhom útoku, škodlivými hrozbami či zraniteľnosťami a bola zabezpečená dôveryhodnosť softvéru pre koncového používateľa.

Účelom tohto dokumentu je zhrnúť a organizácii poskytnúť osvedčené postupy týkajúce sa životného cyklu bezpečného vývoja informačného systému / aplikácie / softvéru (Secure Software Development Life Cycle – SSDLC).

SSDLC možno rozdeliť do nasledovných hlavných fáz:

- Fáza 1: Požiadavky
- Fáza 2: Plánovanie
- Fáza 3: Architektúra a dizajn
- Fáza 4: Vývoj
- Fáza 5: Testovanie
- Fáza 6: Prevádzka
- Fáza 7: Vyraďovanie

Uvedené fázy sú ďalej rozpracované v nasledujúcich kapitolách tohto dokumentu.

2.2 Rozsah platnosti

Tento dokument je platný pre všetkých zamestnancov organizácie a tiež všetky relevantné tretie strany.

3 Fáza 1: Požiadavky

Cieľom tejto fázy SSDLC je identifikovať a definovať príslušné bezpečnostné požiadavky na vyvíjaný softvér.

Bezpečnostné kritériá musi byť začlenené do každej fázy procesu vývoja softvéru, vrátane aplikačnej architektúry a koncepcie použiteľnosti softvérového produktu. Cieľom je explicitne definovať a riešiť bezpečnostné ciele a ciele organizácie a identifikovať požiadavky, ktoré sa vzťahujú na kontext, v ktorom organizácia pôsobí.

V rámci prípravy procesu verejného obstarávania projektu je potrebné zo strany obstarávateľa zabezpečiť, aby súčasťou súťažných podkladov a následne aj návrhu zmluvy s budúcim dodávateľom bola definícia konkrétnych bezpečnostných požiadaviek na aplikácie, ktoré budú predmetom dodania v rámci príslušného projektu. Týmto sa budúci dodávateľ zaviaže, že dodá systém nie len s požadovanými funkčnými vlastnosťami, ktoré definoval budúci biznis vlastník aplikácie alebo systému, ale zároveň aj s požadovanými bezpečnostnými funkciami a vlastnosťami, ktoré musia byť implementované na úrovni informačného systému (ďalej aj „IS“), resp. aplikácie, t. j. zadefinované priamo v zdrojovom kóde alebo musia byť súčasťou dodávky aplikácie alebo systému.

To, aké konkrétne bezpečnostné funkcie budú pre príslušnú aplikáciu alebo systém požadované musí vychádzať najmä z analýzy rizík na základe tzv. „Risk Based Approach“ prístupu a z kategorizácie IS podľa platnej legislatívy, najmä:

- zákona č. 69/2018 Z z. o kybernetickej bezpečnosti a o zmene a doplnení niektorých zákonov v znení neskorších predpisov (ďalej aj „zákon o kybernetickej bezpečnosti“),
- zákona č. 95/2019 o informačných technológiách vo verejnej správe a o zmene a doplnení niektorých zákonov v znení neskorších predpisov (ďalej len „zákon o ITVS“).

V prípade, že IS bude spracúvať aj osobné údaje, je potrebné zohľadniť aj požiadavky zákona č. 18/2018 Z. z. o ochrane osobných údajov a o zmene a doplnení niektorých zákonov v znení neskorších predpisov.

Definícia konkrétnych bezpečnostných požiadaviek a funkcií musí vychádzať v prvom rade z požiadaviek legislatívy pre príslušnú kategóriu novo obstarávaného IS alebo aplikácie.

Okrem legislatívnych požiadaviek je oporúčané pri definovaní konkrétnych bezpečnostných funkcií vychádzať aj z medzinárodných štandardov:

- ISO/IEC 15408 „Information technology — Security techniques — Evaluation criteria for IT security — Part 2: Security functional requirements (Common Criteria)“:

  Tento štandard definuje základný rámec bezpečnostných funkcií, ktoré je potrebné zohľadniť pri definovaní konkrétnych bezpečnostných požiadaviek na základe výsledkov analýzy rizík novo obstarávanej aplikácie, resp. systému, medzi ktoré patria:

  - bezpečnostný audit,
  - komunikácia,
  - kryptografická podpora,
  - nepopierateľnosť (non-repudiation),
  - bezpečnosť používateľských údajov,
  - identifikácia a autentifikácia,

- ochrana súkromia,
- ochrana bezpečnostných funkcií aplikácie,
- prístup do aplikácie.

Detailnejšie informácie ohľadne uvedeného štandardu možno nájsť v prílohe č. 1 tejto metodiky.

- Open Worldwide Application Security Project (OWASP):

  OWASP poskytuje základ pre testovanie technických bezpečnostných opatrení webových aplikácií a tiež poskytuje vývojárom zoznam požiadaviek na bezpečný vývoj. Súbor vybraných OWASP požiadaviek je súčasťou prílohy č. 1 tejto metodiky.

V nasledujúcich podkapitolách sú uvedené hlavné bezpečnostné požiadavky, s ktorými je nevyhnutné pri bezpečnom vývoji softvéru uvažovať.

Na ochranu pred rôznymi typmi hrozieb sa implementujú viaceré vrstvy bezpečnostných opatrení, tzv. Defense in Depth.

Vyhodnotenie bezpečnostných požiadaviek musí byť predmetom formálneho schválenia zo strany organizácie. Finálny výstup musí zároveň slúžiť ako nevyhnutný podklad ku akceptácii vyvíjaného IS pri jeho preberaní do prostredia organizácie.

3.1 Požiadavky na dôvernosť

Riešenie ochrany pred neoprávneným zverejnením údajov, ktoré sú citlivej povahy. Medzi mechanizmy ochrany dôvernosti patria najmä kryptografia a maskovanie.

Požiadavky na dôvernosť je potrebné definovať počas celého životného cyklu informácií od vzniku príslušných údajov až po ich vyradenie. Je potrebné definovať požiadavky na dôvernosť neverejných údajov pri:

- prenose,
- spracovaní,
- ukladaní.

3.2 Požiadavky na integritu

Integrita sa týka nielen ochrany softvéru pred modifikáciou (integrita systému), ale aj údajov, s ktorými softvér pracuje (integrita údajov).

3.3 Požiadavky na dostupnosť

Požiadavky na dostupnosť sú také požiadavky na softvér, ktoré zabezpečujú ochranu pred zničením softvéru a/alebo údajov a predstavujú prevenciu pred odmietnutím služby (DoS) oprávneným používateľom.

3.4 Požiadavky na vývoj webových aplikácií

Požiavky na vývoj webových aplikácií by mali zahŕňať najmä nasledovné požiadavky z pohľadu bezpečnosti:

- aktívneho kódu,
- architektúry webovej aplikácie,
- webovej aplikácie na strane používateľa,
- webovej aplikácie na strane servera.

Okrem vyššie spomínaných požiadaviek je potrebné zohľadniť aj iné všeobecné požiadavky so zameraním na webové aplikácie, ku ktorým patria napr.:

- poznámky vývojárov v kóde aplikácie,
- pravidelná implementácia bezpečnostných „záplat“,
- indexovanie adresárov,
- využívanie vhodných kryptografických algoritmov s primeranými parametrami.

3.5 Požiadavky na organizačné a personálne zabezpečenie

Požiadavky na organizačné a personálne zabezpečenie sú tvorené súborom požiadaviek týkajúcich sa predovšetkým:

- projektových manažérov na oboch stranách projektu,
- osôb zodpovedných za riešenie informačnej bezpečnosti počas projektu implementácie IS,
- osôb zodpovedných za odovzdanie a prevzatie diela
- školení osôb,
- spôsobu umiestnenia a odovzdávania citlivých údajov
- obsahu preberacieho protokolu.

3.6 Rozdelenie zodpovedností a kompetencií

3.6.1 Manažér kybernetickej a informačnej bezpečnosti

Za riadenie a priebežnú kontrolu plnenia bezpečnostných požiadaviek zodpovedá manažér kybernetickej a informačnej bezpečnosti (ďalej aj „MKIB“) alebo ním poverená osoba. Záverečné vyhodnotenie naplnenia bezpečnostných požiadaviek musí byť realizované až vo fáze 5: Implementácia v rámci akceptácie nového IS a jeho nasadzovania do prevádzky.

Hlavnými úlohami MKIB počas jednotlivých fáz SSDLC sú predovšetkým:

- rozpracovanie a riadenie plnenia bezpečnostných požiadaviek,
- vypracovanie a implementácia bezpečnostného projektu a najmä súvisiacej analýzy rizík,
- kooperácia s projektovým manažérom,
- oboznamovanie členov riadiaceho výboru projektu so stavom implementácie bezpečnostných požiadaviek zo strany vývojára,

- povinné akceptácie zo strany MKIB v rámci jednotlivých fáz SSDLC,
- posúdenie bezpečnosti vyplývajúce z vývoja IS prostredníctvom outsourcingu,
- prípadná aktualizácia interných bezpečnostných postupov a interných bezpečnostných smerníc,
- MKIB musí byť zahrnutý do procesu schvaľovania zmien s vplyvom na kybernetickú bezpečnosť,
- a pod.

Zodpovednosť za koordináciu a realizáciu činností súvisiacich s vypracovaním bezpečnostného projektu a analýzy rizík má zástupca dodávateľa zodpovedný za informačnú bezpečnosť vyvíjaného IS v spolupráci s MKIB. Jeho zodpovednosťou je, aby jednotlivé výstupy a činnosti súvisiace s vypracovaním bezpečnostného projektu boli realizované už od začiatku vývoja IS.

Zodpovednosťou MKIB v rámci SSDLC by malo byť aj sledovanie vplyvu novo vyvíjaného IS na interné prostredie organizácie, najmä identifikovanie prípadných zmien v systéme bezpečnostných opatrení a z toho vyplývajúcich zmien na úrovni bezpečnostných smerníc.

3.6.2 Vývojové tímy

Všetky vývojové tímy musia:

- byť informované o spracúvanom bezpečnostnom projekte a prebiehajúcej analýze rizík,
- mať dostatočné vedomosti o bezpečnostných požiadavkách a opatreniach, ktoré musia do výsledného riešenia implementovať,
- poskytovať zástupcovi dodávateľa zodpovednému za informačnú bezpečnosť a MKIB požadovanú súčinnosť.

3.7 Bezpečnostné požiadavky na predmet dodávky

Kategória bezpečnostných požiadaviek na predmet dodávky pozostáva minimálne z požiadaviek na:

- bezpečnostnú architektúru,
- bezpečnostné mechanizmy,
- integráciu IS/aplikácie do existujúceho IT prostredia organizácie,
- spôsob riešenia kontinuity činností predmetu dodávky.

3.8 Zásada najnižších právomocí

Používatelia a zariadenia majú prístup len k tým funkcionalitám v rámci informačného systému, ktoré sú nevyhnutné na plnenie príslušných úloh.

3.9 Oddeľovanie zodpovedností

Jednotliví používatelia nemajú možnosť používať alebo upravovať informačné aktíva bez predchádzajúcej autorizácie. Právomoci používateľov sú obmedzené, aby sa predišlo riziku zneužitia v dôsledku kumulácie.

3.10 Manažment digitálnych identít

V kontexte odporúčaní OWASP je vynucovať používanie silných hesiel, pri implementovanej viacfaktorovej autentizácii, v systéme je zabudovaná správa relácií (ang. „Session management“) a implementovaná je tiež správa cookies.

Autentizačné údaje by mali byť zabezpečené rôznymi faktormi alebo kombináciou týchto faktorov (dvojfaktorová autentizácia – keď sa na overenie využívajú dva faktory, viacfaktorová autentizácia – keď sa na autentifikáciu používajú viac ako dva faktory).

Po úspešnej autentizácii je používateľovi vydaný identifikátor relácie (ID) a toto ID relácie sa používa na sledovanie správania používateľa a udržiavanie stavu autentizácie tohto užívateľa, kým sa relácia neukončí alebo kým sa stav nezmení z autentizovaného na neoverený.

3.11 Požiadavky na správu konfiguračných parametrov

Samotný kód a konfiguračné parametre softvéru musia byť chránené pred útočníkmi.

3.12 Prevádzkové požiadavky

Identifikácia požiadaviek, ako je počet súbežných pripojení k databáze, vzájomné závislosti s inými systémami/aplikáciami a požadované zdroje. Medzi tieto požiadavky patria najmä:

- požiadavky na prostredie,
- požiadavky na archiváciu,
- požiadavky na ochranu proti pirátstvu.

3.13 Požiadavky na ochranu citlivých údajov vrátane osobných údajov

Klasifikácia údajov môže pomôcť pri identifikácii údajov, na ktoré bude potrebné uplatniť zvýšené požiadavky na ich ochranu.

Na citlivé údaje (napr. heslá, osobné údaje, údaje o kreditných kartách, zdravotné záznamy, obchodné záznamy atď.) je aplikovaná vyššia úroveň ochrany, najmä šifrovaná ochrana informácií pri prenose a ukladaní údajov.

Najdôležitejšími údajmi z pohľadu ich citlivosti sú osobné údaje fyzických osôb. Nasledovné pravidlá predstavujú osvedčené postupy v kontexte ochrany osobných údajov:

- nezhromažďovať údaje v prípade, ak nie sú potrebné,
- zhromažďovanie údajov na spracovanie až po tom, ako bol používateľ informovaný o zhromažďovaní údajov a súhlasil s tým,
- zhromažďovanie údajov na spracovanie a uchovávanie so súhlasom používateľa a uchovávanie len počas výslovne stanoveného obdobia uchovávania, ktoré je v súlade s organizačnou politikou a/alebo regulačnými požiadavkami,
- zhromažďovanie údajov a ich uchovávanie bez archivácie, ak údaje prestali byť užitočné a neexistuje požiadavka na uchovávanie.

3.14 Požiadavky na zmluvné ustanovenia

V zmluvných ustanoveniach je kladený dôraz na návrh ustanovení pokrývajúcich bezpečnostné požiadavky a požiadavky na bezpečnosť počas a po skončení zmluvy (SLA) v súlade s §8 vyhlášky Národného bezpečnotného úradu č. 362/2018 Z. z., ktorou sa ustanovuje obsah bezpečnostných opatrení, obsah a štruktúra bezpečnostnej dokumentácie a rozsah všeobecných bezpečnostných opatrení (ďalej aj „vyhláška NBÚ č. 362/2018 Z. z.“).

Medzi zmluvné ustanovenia v zmysle vyššie spomínanej vyhlášky patria najmä tieto:

- na riadenie dodávateľských služieb, akvizície, vývoja a údržby informačných systémov sa pri uzatvorení zmluvy s treťou stranou podľa § 19 ods. 2 zákona o kybernetickej bezpečnosti analyzujú riziká dodávateľských služieb, akvizície, vývoja a údržby informačných systémov spôsobom podľa § 6 vyhlášky NBÚ č. 362/2018 Z. z.,
- vývoj a akvizícia siete a informačného systému základnej služby sa uskutočňuje s ohľadom na zaistenie kompatibility s existujúcimi sieťami a informačnými systémami a zachovanie úrovne bezpečnosti ustanovenej v bezpečnostnej stratégii,
- zmluva s treťou stranou z pohľadu kybernetickej bezpečnosti musí obsahovať najmenej:
  - ustanovenie záväzku tretej strany dodržiavať bezpečnostné politiky prevádzkovateľa základnej služby a vyjadrenie súhlasu s nimi,
  - ustanovenie o povinnosti chrániť všetky informácie poskytnuté prevádzkovateľom základnej služby tretej strane,
  - ustanovenie o povinnosti dodržiavať a prijímať bezpečnostné opatrenia treťou stranou,
  - konkrétnu špecifikáciu a rozsah bezpečnostných opatrení, ktoré prijíma tretia strana a vyjadrenie súhlasu s nimi,
  - konkrétny rozsah činnosti tretej strany,
  - zoznam pracovných rolí tretej strany, ktoré majú mať prístup k informáciám a údajom prevádzkovateľa základnej služby, s povinnosťou oznámiť prevádzkovateľovi základnej služby každú zmenu v personálnom obsadení; osoba zúčastnená na predmete plnenia podpisuje vyjadrenie o zachovávaní mlčanlivosti podľa § 12 ods. 1 zákona,
  - ustanovenie o rozsahu, spôsobu a možnosti vykonávania kontrolných činností a auditu prevádzkovateľom základnej služby v tretej strane,
  - vymedzenie podmienok a možnosti zapojenia ďalšieho dodávateľa úplne alebo čiastočne zabezpečujúceho plnenie pre prevádzkovateľa základnej služby namiesto dodávateľa,
  - ustanovenia o povinnosti informovať prevádzkovateľa základnej služby o kybernetickom bezpečnostnom incidente a o všetkých skutočnostiach majúcich vplyv na zabezpečovanie kybernetickej bezpečnosti,
  - ustanovenia o spôsobe a forme hlásenia ďalších informácií požadovaných prevádzkovateľom základnej služby na plnenie jeho povinností vyplývajúcich zo zákona a ich vymedzenie,
  - ustanovenie o spôsobe a forme hlásenia všetkých informácií majúcich vplyv na zmluvu,

- záväzok tretej strany po ukončení zmluvného vzťahu vrátiť, previesť alebo aj zničiť všetky informácie, ku ktorým má tretia strana počas trvania zmluvného vzťahu prístup prevádzkovateľovi základnej služby,
- záväzok tretej strany po ukončení zmluvného vzťahu udeliť, poskytnúť, previesť alebo postúpiť všetky potrebné licencie, práva alebo súhlasy nevyhnutné na zabezpečenie kontinuity prevádzkovanej základnej služby na prevádzkovateľa základnej služby; tento záväzok tretej strany ostáva v platnosti aj po ukončení zmluvného vzťahu po dobu dohodnutú zmluvnými stranami, ktorá nesmie byť kratšia ako päť rokov po ukončení zmluvného vzťahu.

4 Fáza 2: Plánovanie

Vo fáze plánovania je nevyhnutné, aby organizácia na organizačnej úrovni:

- zriadila špecializovaný bezpečnostný tím fungujúci mimo PMO (projektovej kancelárie pre riadenie projektu) pozostávajúci zo všeobecného odborníka pre oblasť bezpečnosti, architekta pre oblasť bezpečnosti, testera bezpečnosti, príp. iných špecialistov (tieto funkcie môžu byť aj outsourcované),
- jasne zadefinovala úlohy a zodpovednosti pre každú z rolí napr. vo fome RACI matice,
- nastavila taký systém práce, aby bezpečnostný tím fungoval hladko a pomáhal pri prijímaní správnych rozhodnutí smerom k vyvíjanému softvéru,
- schválila a poskytla stratégiu / metodické usmernenie ohľadne testovania bezpečnosti a upresnenie, ako implementovať požiadavky týkajúce sa bezpečnosti (ako, kedy a čo je potrebné otestovať, aké nástroje by sa mali v každej fáze použiť a pod.),
- zabezpečila, aby bol bezpečnostný tím informovaný o akýchkoľvek zmenách v projekte vývoja softvéru.

V rámci tejto fázy sa odporúča, aby organizácia zrealizovala najmenej nasledovné činnosti:

- modelovanie prípadov použitia (Use Case) a prípadov nesprávneho použitia,
- vytvorenie modelu útoku a
- nastavenie procesu riadenia rizík.

4.1 Modelovanie prípadov použitia (Use Case) a prípadov nesprávneho použitia

Cieľom tejto aktivity je identifikácia možného správneho i nesprávneho použitia softvéru a odporučenie bezpečnostných požiadaviek na základe funkčnosti pre vyvinutý softvér.

4.1.1 Analýza scenárov prípadov použitia

Prípad použitia popisuje správanie, ktoré zamýšľal vlastník softvéru. Modelovanie prípadov použitia a vytváranie diagramov je veľmi užitočné pre špecifikáciu požiadaviek. Toto môže byť účinné pri znižovaní nejednoznačných a neúplne formulovaných požiadaviek tým, že sa explicitne presne špecifikuje, kedy a za akých podmienok dochádza k určitému správaniu budúceho softvéru.

Modelovanie prípadov použitia zahŕňa najmä:

- identifikáciu aktérov,
- zamýšľané správanie softvéru (prípady použitia) a
- sekvencie a vzťahy medzi aktérmi a prípadmi použitia.

4.1.2 Analýza scenárov nesprávneho použitia

Prípady nesprávneho použitia naopak modelujú negatívne scenáre. Negatívny scenár je neúmyselné správanie softvéru, ktoré si vlastník softvéru neželá, aby sa vyskytlo v kontexte prípadu použitia. Prípady nesprávneho použitia poskytujú prehľad o hrozbách, ktorých výskyt je relevantný voči vyvíjanému softvéru.

Pri modelovaní prípadov nesprávneho použitia sa modelujú nesprávni aktéri a neúmyselné scenáre alebo správanie. Prípady nesprávneho použitia môžu byť úmyselné alebo náhodné a môžu byť vytvorené prostredníctvom brainstormingu negatívnych scenárov z pozície útočníka.

4.2 Vytvorenie modelu útoku

Vzhľadom na súbor požiadaviek a zoznam hrozieb je úlohou prechádzať zoznam známych útokov a zvážiť, či sa na softvér vzťahuje niektorý útok. Pri vytváraní modelu útoku je postup nasledovný:

- výber vzorov útokov relevantných pre softvér,
- vytvorenie možných prípadov zneužitia so zreteľom na vybrané vzory útokov.

Do modelov je potrebné zahrnúť každú entitu, ktorá môže získať prístup k softvéru, nakoľko model útoku by mať zahŕňať všetky potenciálne zdroje nebezpečenstva pre softvér.

4.3 Nastavenie procesu riadenia rizík

Cieľom procesu riadenia rizík sú nasledovné činnosti:

- korektne cieliť požiadavky na zabezpečenie softvéru, ktorý uchováva, spracúva alebo prenáša informácie,
- umožniť manažmentu prijímať informované rozhodnutia o riadení rizík s cieľom zdôvodniť výdavky, ktoré sú súčasťou rozpočtu na softvér,
- pomáhať manažmentu pri autorizácii softvéru na nasadenie na základe podpornej dokumentácie vyplývajúcej z procesu riadenia rizík.

4.3.1 Hodnotenie rizika

Na určenie rozsahu potenciálnej hrozby a rizika spojeného so softvérom v rámci jeho SSDLC je potrebné identifikovať vhodné kontrolné mechanizmy na mitigáciu identifikovaného rizika.

Metodológia posúdenia rizík zahŕňa deväť primárnych krokov, pričom kroky 2, 3, 4 a 6 možno vykonať paralelne po dokončení kroku 1.

Postup je nasledovný:

Krok 1 – Charakteristika systému

V tomto kroku sa identifikujú hranice softvéru spolu so zdrojmi a informáciami, ktoré tvoria systém. Charakterizácia softvéru stanovuje prístup k analýze a riadeniu rizika, vymedzuje hranice autorizácie pre nasadenie softvéru do prevádzky a poskytuje informácie (napr. hardvér, softvér, konektivitu systému a zodpovednú divíziu alebo podporný personál), ktoré sú nevyhnutné na správne definovanie rizika a jeho úrovne.

Krok 2 – Identifikácia hrozby

Identifikácia potenciálnych zdrojov hrozieb, ktoré sú aplikovateľné pre hodnotený softvér.

Krok 3 – Identifikácia zraniteľnosti

Cieľom identifikácie zraniteľností je vytvoriť zoznam softvérových zraniteľností (chyby alebo slabé miesta), ktoré by mohli zneužiť potenciálne zdroje hrozieb.

Krok 4 – Analýza bezpečnostných opatrení

Analýza bezpečnostných opatrení zahŕňa také opatrenia, ktoré organizácia implementovala alebo plánuje implementovať, aby sa minimalizovala alebo eliminovala pravdepodobnosť využitia softvérovej zraniteľnosti konkrétnou hrozbou.

Krok 5 – Stanovenie pravdepodobnosti

Na odvodenie celkového hodnotenia pravdepodobnosti, s akou môže hrozba využiť potenciálnu zraniteľnosť by sa mali zvážiť minimálne tieto rozhodujúce faktory:

- motivácia a spôsobilosť zdroja hrozieb,
- povaha zraniteľnosti,
- existencia a účinnosť implementovaných bezpečnostných opatrení.

Krok 6 – Analýza dopadu

Na určenie nepriaznivého dopadu vyplývajúceho z úspešného uplatňovania hrozby ohrozenia. Pred začatím analýzy dopadu je potrebné získať nasledujúce informácie:

- zameranie systému (napr. procesy vykonávané softvérom),
- kritickosť systému a údajov (napr. hodnota alebo dôležitosť systému pre organizáciu),
- citlivosť systému a údajov.

Krok 7 – Stanovenie rizika

Stanovenie úrovne rizika pre softvér možno vyjadriť ako:

- pravdepodobnosť, že sa daný zdroj hrozby pokúsi využiť danú zraniteľnosť,
- hodnota dopadu, ak by zdroj hrozby úspešne využil zraniteľnosť,
- reflektovanie existujúcich alebo plánovaných bezpečnostných opatrení na zníženie alebo odstránenie rizika.

Krok 8 – Odporúčania týkajúce sa bezpečnostných opatrení

Cieľom odporúčaných bezpečnostných opatrení je znížiť úroveň rizika spojeného so softvérom na prijateľnú úroveň. Pri odporúčaní bezpečnostných opatrení a alternatívnych riešení na minimalizáciu alebo odstránenie identifikovaných rizík by sa mali zvážiť tieto faktory:

- účinnosť odporúčaných možností (napr. kompatibilita systému),
- legislatíva a regulácia,
- organizačná politika,
- vplyv na prevádzku,

- bezpečnosť a spoľahlivosť.

Krok 9 – Dokumentácia výsledkov

Po dokončení hodnotenia rizík (identifikované zdroje hrozieb a zraniteľnosti, posúdenie rizík a poskytnutie odporúčaných bezpečnostných opatrení), by sa mali výsledky zdokumentovať v oficiálnej správe.

Správa o hodnotení rizika pomáha vedeniu organizácie a vlastníkom rizika prijímať rozhodnutia o relevantných procesoch, rozpočte, prevádzke a zmenách. Na rozdiel od auditnej správy by správa o hodnotení rizika mala byť prezentovaná ako systematický a analytický prístup k hodnoteniu rizika, aby vedenie organizácie pochopilo riziká a vedelo náležite reagovať adresnými bezpečnostnými opatreniami.

4.3.2 Mitigácia rizika

Proces mitigácie rizika slúži na stanovenie priorít, hodnotenie a implementáciu vhodných bezpečnostných opatrení na zníženie rizika odporúčaných v procese hodnotenia rizika.

Možnosti mitigácie rizika

Kvôli nepraktickosti riešiť všetky identifikované riziká, je odporúčané uprednostniť také hrozby a zraniteľnosti, ktoré majú potenciál spôsobiť významný dopad alebo poškodenie. Možnosti na mitigáciu rizika sú:

- akceptovať potenciálne riziko,
- vyhnúť sa riziku,
- obmedziť riziko,
- riadiť riziko vypracovaním plánu na mitigáciu rizika, ktorý určuje priority, implementuje a udržiava kontrolné mechanizmy,
- prenos rizika.

Prístup k implementácii bezpečnostných opatrení

Prístup k implementácii bezpečnostných opatrení pozostáva z nasledujúcich krokov:

1) Stanovenie prioritných krokov
2) Vyhodnotenie odporúčaných možností
3) Vykonanie analýzy nákladov a prínosov
4) Voľba bezpečnostných opatrení
5) Pridelenie zodpovednosti
6) Vypracovanie plánu implementácie ochranných opatrení
7) Implementácia vybraných kontrolných mechanizmov

4.3.3 Pravidelné hodnotenie a posúdenie

Mal by existovať presný harmonogram hodnotenia a posudzovania rizík. Pravidelne vykonávaný proces hodnotenia a posudzovania by mal byť tiež dostatočne flexibilný, aby umožňoval zmeny tam, kde je to opodstatnené (napr. veľké zmeny softvéru v dôsledku zmien vyplývajúcich z nasadenia nových technológií).

5 Fáza 3: Architektúra a dizajn

5.1 Špecificky navrhnutá bezpečnosť

V rámci SSDLC je nevyhnutné zabezpečiť, aby sa primerané technické a organizačné opatrenia prijali už v čase určenia prostriedkov spracúvania, t. j. vo fáze architektúry a dizajnu (tzv. „Security by Design“).

5.2 Štandardná bezpečnosť

Primerané technické a organizačné opatrenia sú trvalo udržateľné, t. j. IT systémy sú predvoleným spôsobom nakonfigurované s bezpečnými nastaveniami a voľbami (tzv. „Security by Default“).

5.3 Dizajn základnych bezpečnostných prvkov

Cieľom je navrhnúť softvér tak, aby riešil najmä základné bezpečnostné prvky dôvernosti, integrity, dostupnosti, autentizácie, autorizácie a auditu.

5.3.1 Dizajn dôvernosti

Zabezpečiť ochranu pred zverejnením možno viacerými spôsobmi pomocou kryptografických a maskovacích techník. Maskovanie je užitočné na ochranu pred zverejnením, keď sa údaje zobrazujú na obrazovke alebo na tlačených formulároch.

Kryptografia slúži na zabezpečenie dôvernosti, používa sa pri prenose údajov alebo ich ukladaní do dátových skladov alebo offline archívov. Pracovný faktor, ktorý zahŕňa kryptografickú ochranu je závislý na:

- veľkosti kľúčov,
- správe kľúčov,
- rotácii (výmene) kľúčov.

Šifrovacie algoritmy sú primárne dvoch typov:

- symetrické algoritmy – používajú jeden kľúč na šifrovanie a dešifrovanie, ktorý je zdieľaný medzi odosielateľom a príjemcom,
- asymetrické algoritmy – používajú dva kľúče, z ktorých jeden sa musí držať v tajnosti a označuje sa ako súkromný kľúč, zatiaľ čo druhý kľúč sa prezradí každému, s kým je potrebné uskutočniť bezpečnú komunikáciu a transakcie. Kľúč, ktorý je verejne vystavený, sa nazýva verejný kľúč.

5.3.2 Dizajn integrity

Cieľom je zabezpečiť, že nedôjde k žiadnej neoprávnenej úprave softvéru alebo údajov pomocou niektorej z nasledujúcich techník alebo ich kombinácie:

- hašovanie – slúži na zabezpečenie integrity dát; na základe hašu je potom možné identifikovať zmeny v údajoch počas prenosu alebo uloženia,
- referenčná integrita – mechanizmus, ktorý zabezpečuje konzistenciu údajov, najmä v relačných databázových systémoch (RDBMS). Využíva primárne kľúče a príslušné cudzie kľúče v databáze

na zabezpečenie integrity údajov.

- uzamknutie zdrojov – je technika, ktorá zabraňuje súbežným operáciám nad rovnakým objektom (napríklad záznamom v databáze).

5.3.3 Dizajn dostupnosti

Keďže ide o fázu dizajnu, je odporúčané zvážiť požiadavky na konfiguráciu. Na zabezpečenie dostupnosti sa používajú tieto techniky:

- replikácia,
- failover (preklopenie pri poruche),
- škálovateľnosť.

5.3.4 Dizajn autentifikácie

Pri určení typu autentifikácie, ktorý sa vyžaduje podľa špecifikácie v dokumentácii požiadaviek, je potrebné zvážiť použitie viacfaktorovej autentizácie a jednotného prihlásenia (SSO).

Odporúčané použitie viacfaktorovej autentizácie poskytuje zvýšenú bezpečnosť.

Ak je potrebné implementovať SSO, pri ktorom sa raz overí potvrdená identita používateľa a overené prístupové oprávnenia sa odovzdávajú ďalším systémom alebo aplikáciám, potom je nevyhnutné zohľadniť pri návrhu softvéru vplyv tohto mechanizmu na bezpečnosť.

5.3.5 Dizajn autorizácie

Pri implementácii autorizácie je potrebné zvážiť, aký typ autorizácie bude použitý v súlade s požiadavkami, či už ide o autorizáciu na základe rolí alebo autorizáciu založenú na zdrojoch. V prípade autorizácie na základe rolí je nevyhnutné zabezpečiť, aby neboli prítomné konfliktné roly, ktoré by mohli obchádzať zásadu oddelenia povinností. To znamená, že jednotlivý používateľ by nemal byť súčasne priradený do rolí, ktoré by mu umožňovali vykonávať nezlučiteľné funkcie. Tento aspekt je dôležitý pre zabezpečenie integrity a bezpečnosti systému.

Pri návrhu autorizácie je možné použiť správu oprávnení, ktorá umožňuje kontrolu prístupu. Tento prístup umožňuje detailné nastavenie oprávnení pre jednotlivé zdroje a činnosti. To zabezpečuje, že používatelia majú prístup len k tým zdrojom a operáciám, ktoré sú pre ich úlohu a povinnosti relevantné. Tým sa minimalizuje riziko neoprávneného prístupu a zneužitia oprávnení.

Pri návrhu autorizácie je nevyhnutné zohľadniť nielen bezpečnostné aspekty, ale aj výkonnostné požiadavky systému. Je dôležité dosiahnuť vyvážený kompromis medzi zabezpečením dát a efektívnym vykonávaním operácií.

Mechanizmus riadenia prístupu vhodný pre všeobecné objekty nešpecifikovaných typov je nasledovný:

Adresár

Jedným z jednoduchých spôsobov ochrany je použitie mechanizmu, ktorý funguje ako adresár súborov. Každý súbor má jedinečného vlastníka, ktorý má „kontrolné“ prístupové práva (vrátane práva deklarovať, kto má aký prístup) a kedykoľvek zrušiť prístup ktorejkoľvek osobe. Každý používateľ má adresár súborov, v ktorom sú uvedené všetky súbory, ku ktorým má daný používateľ prístup.

Zoznam kontroly prístupu (ACL)

Pre každý objekt existuje jeden takýto zoznam, v ktorom sú uvedené všetky subjekty, ktoré majú mať prístup k objektu, a ako je definovaný je ich prístup. Tento prístup sa líši od zoznamu adresárov, pretože pre každý objekt existuje jeden zoznam na riadenie prístupu.

Matica riadenia prístupu

Tabuľka, v ktorej každý riadok predstavuje subjekt, každý stĺpec predstavuje objekt a každá položka je súbor prístupových práv daného subjektu k danému objektu.

5.3.6 Logovanie

V rámci SSDLC je zabezpečiť auditné záznamy softvéru najmä v prípade narušenia, predovšetkým na forenzné účely. Auditné záznamy by mali obsahovať aspekty „kto“, „čo“, „kde“ a „kedy“ softvérových operácií. V rámci „kto“ je dôležité nezabudnúť aj na dávkové procesy a služby.

5.4 Bezpečné rámce

V rámci SSDLC musí byť zabezpečené, že sa pri vývoji softvéru použijú iba najnovšie a najbezpečnejšie verzie nástrojov a komponentov na vývoj softvéru. Je potrebné použiť iba také softvérové knižnice a komponenty, ktoré sú aktívne podporované, majú pozitívne referencie a pochádzajú od dôveryhodných dodávateľov.

Určenie programovacieho jazyka, ktorý bude použitý na implementáciu dizajnu, so sebou prináša riziká a bezpečnostné výhody. Existujú dva hlavné typy programovacích jazykov:

- vysokoúrovňové programovacie jazyky – sú navrhnuté tak, aby boli ľahko čitateľné a zrozumiteľné pre ľudí. Poskytujú vyššiu úroveň abstrakcie a sú bližšie k prirodzenému jazyku. Vysokoúrovňové jazyky sú zvyčajne jednoduchšie na naučenie a použitie, pretože abstrahujú mnoho detailov na nižšej úrovni.
- nízkoúrovňové programovacie jazyky – sú bližšie k hardvéru a poskytujú priamu kontrolu nad počítačovým systémom. Sú často používané pre úlohy, ktoré vyžadujú jemné ovládanie systémových prostriedkov alebo optimalizáciu výkonu. Nízkoúrovňové jazyky sú zložitejšie a vyžadujú hlbšie porozumenie architektúre počítača.

5.5 Typ údajov, formát, rozsah a dĺžka

Cieľom je zabezpečiť integritu s typom údajov, formátom, rozsahom a dĺžkou z hľadiska dizajnu:

- primitívne alebo vstavané dátové typy sú ako Character, Integer, Floating-Point number a Boolean,
- používateľom definované dátové typy sa z bezpečnostného hľadiska neodporúčajú, pretože potenciálne zvyšujú možnosť útoku,
- množina hodnôt a povolené operácie s touto množinou hodnôt, ktoré sú definované dátovým typom.

5.6 Dizajn rozhrania

Pri vytváraní softvéru je potrebné uplatniť úvahy o návrhu rozhrania, ako napríklad:

- používateľské rozhranie,
- aplikačné programové rozhrania (API),

- rozhrania na správu bezpečnosti (SMI),
- rozhranie mimo pásma,
- log rozhrania.

5.7 Prepojenie

Vo väčšine mobilných aplikácií je ochrana údajov na klientovi ponechaná na samotnú aplikáciu, a preto by aplikácie mali byť navrhnuté tak, aby sa vyhli ukladaniu akýchkoľvek citlivých informácií v samotnom prostredí sandboxu aplikácie. Na zabezpečenie dôvernosti môže byť potrebné zvážiť vydavateľov a API tretích strán, ktoré poskytujú kryptografické služby (šifrovanie a dešifrovanie).

Ak sú údaje uložené na mieste v sieti, malo by byť chránené aj sieťové úložné zariadenie (NAS). Mali by sa navrhnúť len overené a autorizované pripojenia k NAS, a ak je prístup navrhnutý tak, aby bol obmedzený rozsahom IP, je potrebné venovať osobitnú pozornosť hrozbám falšovania IP.

6 Fáza 4: Vývoj

Cieľom tejto fázy SSDLC je zabezpečiť bezpečnostnú kontrolu definovanú v bezpečnostných požiadavkách, pričom je potrebné aplikovať technologické a procesné aspekty kódovania a zabrániť hrozbám identifikovaným v procese riadenia rizík počas procesu kódovania.

6.1 Najpoužívanejšie spôsoby vývoja

V rámci tejto kapitoly sú popísané tri najpoužívanejšie spôsoby vývoja systémov:

- Waterfall,
- Agile,
- DevOps.

6.1.1 Waterfall

Waterfall je tradičná, plánom riadená prax pre vývoj systémov. Ide o jeden z prvých prístupov k životnému cyklu vývoja softvéru (SDLC). Waterfall sa najlepšie aplikuje na projektoch vývoja softvéru, ktoré sú dobre definované, predvídateľné a je nepravdepodobné, že sa výrazne zmenia. Zvyčajne sa to týka jednoduchších projektov menšieho rozsahu. Jeho sekvenčný charakter spôsobuje, že do značnej miery nereaguje na úpravy, takže rozpočty a termíny dodávok budú ovplyvnené, keď sa obchodné požiadavky zmenia počas vývojového cyklu.

Charakteristickým pre tento prístup je postup vývoja v etapách, konkrétne:

- Fáza 1: Plánovanie,
- Fáza 2: Analýza,
- Fáza 3: Návrh,
- Fáza 4: Implementácia,
- Fáza 5: Testovanie,
- Fáza 6: Nasadenie do prevádzky.

Výstup z jednej etapy je potrebný na spustenie ďalšej etapy.

SSDLC popísaný v rámci tejto metodiky sa pri waterfall spôsobe vývoja aplikuje veľmi jednoducho, nakoľko fázy SSDLC a waterfallu sú si veľmi podobné a ich napárovanie je triviálne.

6.1.2 Agile

Agilné postupy vývoja softvéru vznikli na základe potreby prispôsobiť sa a rýchlo dodať produkt. Agilná metodológia umožňuje skúmanie nových nápadov a rýchlejšie rozhodovanie o tom, ktoré z týchto nápadov sú životaschopné.

Agilné metódy sú navyše navrhnuté tak, aby sa prispôsobili meniacim sa obchodným potrebám počas vývoja. V Agile sa používajú primárne dva rámce:

- Scrum,
- Kanban.

Kľúčovými komponentmi Scrumu sú iterácie a rýchlosť, zatiaľ čo kľúčovým bodom Kanbanu je tzv. work in progress.

Agilné metódy sú založené na iteratívnom, postupnom vývoji, ktorý umožňuje rýchlo priniesť životaschopný obchodný produkt. Postupný vývoj rozdeľuje produkt na menšie časti; každá časť sa vytvára, hodnotí a prispôsobuje. Agilné projekty nezačínajú s úplnými definíciami vopred; očakáva sa variabilita v priebehu vývoja. Dôležitá je neustála spätná väzba, ktorá je súčasťou procesu vývoja.

Základnými fázami agilného vývoja aplikácií sú:

- Fáza 1: Plánovanie,
- Fáza 2: Analýza,
- Fáza 3: Návrh,
- Fáza 4: Implementácia,
- Fáza 5: Testovanie.

Aj v prípade Agile je možné využiť túto metodiku jednoduchým napárovaním fáz Agile a tu popísanými pravidlami a požiadavkami SSDLC.

6.1.3 DevOps

DevOps je metodika v oblasti vývoja softvéru. Používa sa ako súbor postupov a nástrojov, ktorý integruje a automatizuje vývoj softvéru (Dev) a IT operácie (Ops) ako prostriedok na zlepšenie a skrátenie životného cyklu vývoja systémov. DevOps je doplnkom k agilnému vývoju softvéru, ktorý umožňuje predovšetkým:

- rýchle dodanie hotového kódu,
- minimálne poruchy a
- okamžité zotavenie sa z porúch.

DevOps je primárne o zlúčených tímoch a automatizácii v agilnom vývoji. Kultúra DevOps vyrástla z Agile a pomáha urýchliť uvedenie kódu/produktu do prevádzky. Použité postupy DevOps sa odvíjajú primárne od:

- veľkosti projektu,
- zručností členov tímu,
- voľnosti organizačnej štruktúry a schopnosti organizácie upravovať štruktúru podľa potreby,
- cloud vs. on-premise architektúre,
- rozpočtu na projekt,
- a pod.

Hlavnými fázami DevOps spôsobu vývoja sú:

- Fáza 1: Neustály vývoj,
- Fáza 2: Priebežné testovanie,
- Fáza 3: Nepretržitá integrácia,

- Fáza 4: Nepretržité nasadenie,
- Fáza 5: Nepretržité monitorovanie,
- Fáza 6: Nepretržitá spätná väzba,
- Fáza 7: Nepretržité operácie.

Pri tomto spôsobe vývoja je aplikovanie SSDLC komplikovanejšie; namapovanie jednotlivých fáz SSDLC na DevOps fázy však je možné:

DevOps fáza	SSDLC fáza
Fáza 1: Neustály vývoj	Fáza 1: Požiadavky Fáza 3: Architektúra a dizajn Fáza 4: Vývoj
Fáza 2: Priebežné testovanie	Fáza 6: Testovanie
Fáza 3: Nepretržitá integrácia	Fáza 1: Požiadavky Fáza 5: Implementácia
Fáza 4: Nepretržité nasadenie	Fáza 5: Implementácia
Fáza 5: Nepretržité monitorovanie	Fáza 7: Prevádzka
Fáza 6: Nepretržitá spätná väzba	Fáza 2: Plánovanie
Fáza 7: Nepretržité operácie	Fáza 7: Prevádzka Fáza 8: Vyraďovanie

6.1.4 Postupy v zmysle vyhlášky ÚPVII č. 85/2020 Z. z.

Vyhláška Úradu podpredsedu vlády Slovenskej republiky pre investície a informatizáciu č. 85/2020 Z. z. o riadení projektov v znení neskorších predpisov (ďalej aj „vyhláška ÚPVII č. 85/2020 Z. z.“) okrem iného ustanovuje aj štandardy projektového riadenia podľa § 24 ods. 1 písm. j) zákona o ITVS a podrobnosti o riadení projektov informačných technológií verejnej správy.

V rámci vyhlášky ÚPVII č. 85/2020 Z. z. je fázovanie životného cyklu projektu ITVS nasledovné:

- prípravná fáza projektu, ktorej účelom je vypracovanie rámcového projektového zámeru, rámcového zoznamu rizík a závislostí, zriadenie riadiaceho výboru projektu a vytvorenie predpokladov na iniciáciu projektu,
- iniciačná fáza projektu na vypracovanie projektového produktu Projektový zámer, Zoznam rizík a závislostí, BC/CBA – odôvodnenie projektu s katalógom funkčných, nefunkčných a technických požiadaviek, Prístup k projektu a príprava aspektov projektu a manažérskych produktov na realizačnú fázu projektu,
- realizačná fáza projektu, ktorej účelom je navrhnúť, vytvoriť, otestovať, dodať a nasadiť špecializované produkty a ktorá sa realizuje v týchto etapách:
  - Analýza a Dizajn,
  - Nákup technických prostriedkov, programových prostriedkov a služieb,

- Implementácia a Testovanie,
- Nasadenie a Postimplementačná podpora,

- dokončovacia fáza projektu, ktorej účelom je po úplnom dodaní všetkých špecializovaných produktov administratívne aj technicky uzavrieť celý projekt.

SSDLC popísaný v tejto metodike sa páruje na fázovanie podľa vyhlášky ÚPVII č. 85/2020 Z. z. analogicky.

6.1.5 CI/CD

Kontinuálne nasadenie (CD) kódu aplikácie úzko súvisí s jeho kontinuálnou integráciou (CI) a týka sa uvoľnenia softvéru, ktorý prejde automatizovanými testami do produkcie. V situácii, v ktorej došlo k dodávke istej časti kódu, je nutné nasadiť tento kód úspešne do produkcie.

Každý z postupov a krokov, ktoré so sebou CI/CD pipeline prináša, je možné vykonať manuálne, ich skutočná hodnota sa však realizuje automatizáciou. Automatizovaná CI/CD pipeline sa sústredí na:

- tvorbu kódu aplikácie,
- spúšťanie testov,
- bezpečné nasadzovanie nových verzií aplikácií do produkcie
- odstraňovanie chýb z kódu.

Hlavnými fázami CI/CD pipeline sú:

- Fáza 1: Plánovanie,
- Fáza 2: Kódovanie,
- Fáza 3: Build,
- Fáza 4: Testovanie,
- Fáza 5: Release,
- Fáza 6: Nasadenie,
- Fáza 7: Prevádzkovanie.

6.1.6 Mikroslužby

Mikroslužby (mikroservisy) je pojem z oblasti vývoja softvéru. Ide o jeden z variantov softvérovej architektúry orientovanej na služby (SOA), kde sú aplikácie definované ako súbor voľne previazaných služieb.

Mikroservisná architektúra sa často používa pre cloud natívne aplikácie a aplikácie využívajúce nasadzovanie pomocou tzv. „lightweight“ kontajnerov. Mikroslužby môžu byť ľahko individuálne škálované v porovnaní so službami v monolitickej aplikácii; pri mikroslužbách je potrebné škálovať iba konkrétne mikroslužby, ktoré majú vyššie nároky na zdroje, čo poskytuje výhodu optimalizácie zdrojov a nákladov.

6.1.7 Kontajnerizácia (Kubernetes)

Kubernetes predstavuje riešenie, ktoré umožňuje ľahké nasadenie nových aplikácií, resp. upraviť už existujúce. Pomáha riešiť škálovateľnosť, dostupnosť aplikácie, jej životný cyklus a tým robí aplikácie efektívne a dobre prispôsobiteľné.

Kubernetes je otvorená (open source) platforma, ktorá automatizuje operácie s kontajnermi. Odstraňuje mnoho ručných procesov, ktoré sa týkajú nasadenia a zjednodušovania kontajnerových aplikácií. Z tohto dôvodu je Kubernetes ideálnou platformou pre hosťovanie aplikácií v cloude, ktoré vyžadujú rýchle škálovanie a časté upravovanie.

6.2 Všeobecné best practices pre spôsoby vývoja

Pre všetky spôsoby vývoja popísané v bode 6.1 tejto metodiky sa odporúča reflektovať najmä nasledovné osvedčené princípy bezpečného vývoja softvéru:

- určenie požiadaviek – bezpečnostné kritériá začlenené do každej fázy procesu vývoja softvéru, vrátane aplikačnej architektúry a koncepcie použiteľnosti softvérového produktu,
- hĺbková ochrana – na ochranu pred rôznymi typmi hrozieb implementovať viaceré vrstvy bezpečnostných opatrení,
- zásada najnižších právomocí – používatelia a zariadenia majú prístup len k tým funkcionalitám v rámci informačného systému, ktoré sú nevyhnutné na plnenie príslušných úloh,
- oddeľovanie zodpovedností – jednotliví používatelia nemajú možnosť používať alebo upravovať informačné aktíva bez predchádzajúcej autorizácie, právomoci používateľov sú obmedzené, aby sa predišlo riziku zneužitia v dôsledku kumulácie,
- špecificky navrhnutá bezpečnosť – primerané technické a organizačné opatrenia sú prijaté už v čase určenia prostriedkov spracúvania, t. j. už vo fáze návrhu a vývoja IT systémov,
- štandardná bezpečnosť – primerané technické a organizačné opatrenia sú trvalo udržateľné, t. j. IT systémy sú predvoleným spôsobom nakonfigurované s bezpečnými nastaveniami a voľbami,
- riadenie zraniteľností – posudzovaním a záplatami je minimalizovaný počet známych zraniteľností a potenciálnych slabých miest v IT systémoch a infraštruktúre,
- logovanie – správne nakonfigurovaný proces logovania bezpečnostných informácií napomáha včas zachytiť neobvyklé správanie softvéru, a systémové udalosti skôr, ako sa rozvinú do skutočného incidentu,
- bezpečné rámce – používajú sa iba najnovšie a najbezpečnejšie verzie nástrojov a komponentov na vývoj softvéru, softvérové knižnice komponenty sú aktívne podporované, majú pozitívne referencie a pochádzajú od dôveryhodných dodávateľov,
- validácia vstupov – je správne nastavená stratégia syntaktického a sémantického overovania vstupov, kód je konzistentný a softvérové moduly neakceptujú žiadne nesprávne a neočakávané vstupy,
- oddelené prostredia – vývoj prebieha mimo produkčného prostredia a bez použitia "ostrých dát", s definovaným a riadeným procesom správy zdrojového kódu a verzií,

- manažment digitálnych identít – v kontexte odporúčaní OWASP je zaručené používanie silných

hesiel, je implementovaná viacfaktorová autentifikácia, v systéme je zabudovaná správa relácií a implementovaná je správa cookies,

- ochrana citlivých údajov – na citlivé údaje (napr. heslá, osobné údaje, údaje o kreditných kartách, zdravotné záznamy, obchodné záznamy atď.) je aplikovaná vyššia úroveň ochrany, najmä šifrovaná ochrana informácií pri prenose a ukladaní údajov,
- spracovanie výnimiek a chýb – systém sa nezrúti ani pri výskyte chyby a spôsob, akým bude aplikácia reagovať na množstvo nepredvídateľných stavov sa opiera o súbor účinných núdzových scenárov,
- penetračné testovanie – potenciálne zraniteľné miesta systému vyplývajúce z chýb kódovania, alebo chýb konfigurácie sú odhalené testovaním, ešte pred nasadením do produkčného prostredia,
- súlad – systém je navrhnutý tak, aby počas jeho budúcej prevádzky boli dodržané príslušné právne predpisy, požiadavky odvetvovej regulácie a technické normy týkajúcich sa informačnej bezpečnosti.

6.3 Ďalšie špecifické bezpečnostné úlohy vo fáze vývoja

6.3.1 Bežné softvérové zraniteľnosti a bezpečnostné opatrenia

Cieľom je identifikovať najčastejšie slabé miesta, ktoré sú výsledkom nezabezpečeného kódovania a aplikovať bezpečnostné opatrenia.

6.3.1.1 Databáza zraniteľností

Organizácia by mala pravidelne preverovať databázy zraniteľností alebo úložiská známych zraniteľností, ktoré boli zistené ako dôsledok nedostatkov a chýb a bugov v implementovanom softvéri. Organizácia sa však môže obrátiť na aktuálny zoznam zraniteľností softvéru (napríklad projekt OWASP Top 10 alebo Common Weakness Enumeration).

6.3.1.2 Defenzívne kódovacie praktiky

Rozpoznanie, vyhodnotenie a zníženie rozsahu útoku softvérového kódu. Niektoré príklady zníženia rozsahu útoku týkajúce sa kódu sú:

- zníženie množstva kódu a služieb, ktoré sa štandardne vykonávajú,
- zníženie objemu kódu, ku ktorému majú prístup nedôveryhodní používatelia,
- obmedzenie škôd pri zneužití kódu.

6.3.2 Bezpečné softvérové procesy

Cieľom je zabezpečiť bezpečnosť softvéru vykonaním určitých procesov ako doplnok ku samotnému kódovaniu.

6.3.2.1 Verzia zdrojového kódu

Zabezpečenie toho, aby vývojový tím pracoval so správnou verziou kódu. Poskytuje možnosť vrátiť sa k predchádzajúcej verzii v prípade potreby. Okrem toho poskytuje možnosť sledovať vlastníctvo a zmeny kódu a na účely aktualizácie verzie.

6.3.2.2 Analýza kódu

Vykonanie automatizovaného procesu kontroly kvality kódu a slabých miest, ktoré možno zneužiť. Vykonáva sa predovšetkým dvoma spôsobmi: statickým a dynamickým.

Statická analýza kódu zahŕňa kontrolu kódu bez spustenia kódu (alebo softvérového programu). Túto analýzu zvyčajne vykonáva tretia strana.

Dynamická analýza kódu je kontrola kódu počas spustenia kódu. Túto analýzu zvyčajne vykonáva vývojár.

6.3.2.3 Kontrola kódu

Vykonanie manuálneho systematického hodnotenia zdrojového kódu s cieľom zistiť syntaktické problémy a slabé miesta v kóde, ktoré môžu ovplyvniť výkonnosť a bezpečnosť softvéru. Takáto kontrola sa zvyčajne vykonáva medzi vývojármi. Sémantické problémy, ako je logika a chyby v návrhu, sa pri preskúmaní kódu zvyčajne nezisťujú, ale preskúmanie kódu sa môže použiť na overenie modelu hrozieb vytvoreného vo fáze návrhu projektu vývoja softvéru.

6.3.2.4 Testovanie vývojára

Vykonávanie zámerného a systematického zamestnávania vývojárov testovacími nástrojmi a technikami s cieľom vytvoriť testovateľný a udržiavateľný softvér s čo najmenším počtom chýb. Vývojárske testovanie si vyžaduje štruktúrovaný prístup a zvládnutie niekoľkých základných kompetencií, z ktorých najdôležitejšie sú pochopenie faktorov testovateľnosti, základné techniky testovania a jednotkové testovanie.

Bežné typy testov pre aplikácie sú najmä:

- jednotkové testy,
- integračné testy,
- regresné testy,
- softvérové testy.

6.3.3 Nastavenie prostredí

V rámci SSDLC je nevyhnutné zabezpečiť, aby vývoj softvéru prebiehal mimo produkčného prostredia a bez použitia produkčných dát, s definovaným a riadeným procesom správy zdrojového kódu a verzií. Ďalej je dôležité zabezpečiť, aby boli potrebné parametre na spustenie softvéru vhodne nakonfigurované, aby vývojové a testovacie prostredie zodpovedalo konfiguračnému nastaveniu produkčného prostredia a aby simulačné testovanie identicky emulovalo nastavenia (vrátane reštriktívnych nastavení) prostredia, v ktorom bude softvér nasadený po akceptácii.

6.3.4 Zabezpečenie vývojárskeho prostredia

Cieľom je chrániť zdrojový kód pred prístupom osoby, ktorá nie je spojená s projektom počas fázy vývoja a zabezpečiť integritu vytvoreného zdrojového kódu. Do zabezpečenia vývojárskeho prostredia zaraďujeme aj fyzické zabezpečenie prístupu k vývojárskemu prostrediu.

6.3.4.1 Používanie zoznamov riadenia prístupu (ACL)

Zoznamy riadenia prístupu (ACL) zabraňujú prístupu neoprávneným používateľom.

6.3.4.2 Používanie softvéru na správu verzií

Softvér na správu verzií zabezpečí, že vytvorený kód bude mať správnu verziu.

6.3.4.3 Vytvorenie automatizácie

Automatizácia preberá manuálne činnosti, ktoré denne vykonávajú členovia tímu a automatizuje ich. Niektoré z týchto činností zostavovania zahŕňajú kompiláciu zdrojového kódu,nasadenie a inštaláciu. Keď sa na automatizáciu procesov používajú skripty, je dôležité zabezpečiť, aby sa pri používaní týchto skriptov neobchádzali bezpečnostné opatrenia.

6.3.4.4 Rutina podpisovania kódu

Hlavnou úlohou podpisovania kódu (tzv. „CodeSign") je zabezpečiť integritu vyvíjaného zdrojového kódu.

6.3.5 Bezpečnosť používania vývojárskych softvérových nástrojov

Pri využívaní vývojárskych softvérových nástrojov je nutné dodržiavanie najmä nasledovných podmienok:

- môžu byť použité iba vývojárske softvérové nástroje získané legálnym spôsobom,
- musia byť stále podporované výrobcom vývojárskeho softvérového nástroja (t. j. výrobca poskytuje bezpečnostné aktualizácie) a nesmú byť označené ako nepodporované,
- môžu byť použité iba dôveryhodné (a zároveň rozšírené) frameworky/knižnice, ktoré kladú dôraz na bezpečnosť a predchádzanie bežným programátorským chybám a zároveň rýchlo zverejňujú opravy bezpečnostných chýb,
- musia byť pravidelne aplikované bezpečnostné záplaty vydané výrobcom vývojárskeho nástroja,
- musia byť implementované príslušné opatrenia na zabezpečenie integrity vyvíjaného IS na základe najvyššej požadovanej úrovne ochrany dôvernosti, integrity a dostupnosti informácií, ktoré budú spracovávané vo vyvíjanom riešení,
- na prístup do úložiska zdrojového kódu je potrebné vyžadovať politiku prístupu založenú na RBAC (Role based access control),
- všetky zmeny uskutočnené v zdrojovom kóde musia byť predmetom zaznamenávania a preskúmavania,
- v prípade používania vývojárskeho softvérového nástroja na paralelný vývoj viacerých IS musia byť jednotlivé projekty IS v tomto nástroji bezpečne oddelené.

6.3.6 Bezpečnosť vyvýjaného zdrojového kódu

Vyvíjaný zdrojový kód by mal byť vytváraný s ohľadom na bezpečné metódy programovania vychádzajúce z požiadaviek štandardov a noriem v tejto oblasti bezpečnosti (napr. OWASP Secure

Coding Practices, CERT Secure Coding Standards, Microsoft Secure Coding Guidelines). Konkretizácia týchto štandardov závisí od charakteristík vyvíjaného softvéru, využívaných programovacích jazykov, platforiem či zaužívaných postupov dodávateľov a musí byť vykonaná na začiatku vývoja.

6.3.7 Využitie dát počas vývoja IS

Počas vývojovej fázy vývoja IS by nemali byť použité prevádzkové dáta, osobné údaje prípadne iné citlivé informácie. V prípade potreby vytvorenia testovacích dát z produkčných dát je potrebné použiť vhodné formy a metódy „anonymizácie“ týchto údajov, ktoré zabezpečia, že nebude možné získať žiadne, ani „postranné“ alebo inak odvodené informácie z týchto dát. Zároveň je potrebné zvýšenú pozornosť venovať aj zabezpečeniu prostredia, v ktorom bude „anonymizácia“ vykonaná.

V prípade, že jediná vhodná a efektívna možnosť testovania IS je s produkčnými prevádzkovými dátami, vývojové, resp. testovacie prostredie je nutné zabezpečiť rovnako, ako bude zabezpečené finálne produkčné prostredie.

6.3.8 Vývoj IS prostredníctvom outsourcingu

Pri vývoji IS prostredníctvom outsorcingu je odporúčané venovať sa nasledujúcim oblastiam a činnostiam:

- oboznámenie dodávateľov o bezpečnostných požiadavkách vyplývajúcich z interných predpisov organizácie,
- bezpečnostné povedomie, školenia a poučenia zamestnancov dodávateľa vyvíjaného IS,
- bezpečné riadenie prístupu dodávateľov do siete a k aktívam organizácie,
- bezpečnosť vývojového prostredia IS,
- štandardy a požiadavky pre bezpečné programovanie a vývoj IS.

6.3.9 Hardening systému

Úlohou tejto techniky je nastaviť minimálnu (najreštriktívnejšiu) úroveň zabezpečenia softvéru.

V tejto súvislosti je tiež dôležité posilniť hostiteľský operačný systém pomocou aktualizácií a záplat. Hardening softvéru zahŕňa nastavenie konfiguračných nastavení softvéru tak, aby bol štandardne bezpečný.

6.3.10 Tvorba zdrojovéjo kódu

Pri vytváraní zdrojového kódu je od dodávateľov nevyhnutné vyžadovať dodržiavanie základných pravidiel kódovania a správy zdrojových kódov.

Medzi takéto základné pravidlá patrí napr.:

- používanie bezpečných návrhových vzorov,
- pravidelná kontrola zdrojového kódu,
- riadené verzionovanie,
- vykonávanie validácie vstupov,
- spracovanie a zaznamenávanie chýb,
- realizácia bezpečnej konfigurácie (hardeningu).

7 Fáza 5: Testovanie

Cieľom fázy SSDLC je overiť funkčnosť a bezpečnosť softvéru pomocou testovania zabezpečenia kvality a zabezpečiť, aby vyvinutý kód bežal podľa plánu

7.1 Správa testovacích dát

Cieľom je identifikácia vstupných testovacích údajov, ktoré sa očakávajú na výstupe po normálnej prevádzke softvéru.

7.1.1 Identifikácia výstupných testovacích údajov

Identifikácia očakávaných výstupných testovacích údajov pomáha potvrdiť, či softvér spĺňa požiadavky. Kvalita testovacích údajov priamo súvisí s kvalitou samotného testu, a preto je potrebné testovacie údaje riadiť. Produkčné údaje by sa nikdy nemali importovať do testovacích prostredí a spracovávať sa v nich. Dôrazne sa odporúča používať fiktívne údaje.

7.1.2 Aplikácia testovania so syntetickými transakciami

Vykonávanie transakcií, ktoré nemajú žiadnu obchodnú hodnotu súvisiacu s fiktívnymi údajmi. Tzv. syntetické transakcie môžu byť pasívne alebo aktívne. Pasívne syntetické transakcie sa neukladajú (ani neudržiavajú) a nemajú žiadny zostatkový vplyv na samotný softvér. Používanie aktívnych syntetických transakcií si vyžaduje venovať pozornosť nastaveniu údajov a prostredia takým spôsobom, aby to nemalo vplyv na produkčné prostredie.

7.1.3 Testovanie riešení správy údajov

Môže pomôcť pri vytváraní referenčne celých podskupín údajov o produkcii. Znížiť niektoré problémy, ktoré súvisia s vytváraním kvalitných testovacích údajov a ich správou v testovacích prostrediach. Tieto riešenia automaticky zisťujú dátové vzťahy analýzou a zachytením atribútov tabuliek. Je potrebné zabezpečiť, aby pravidlá extrakcie zohľadňovali úložný priestor, ktorý je k dispozícii v testovacom prostredí, a aby proces extrakcie neskončil extrakciou veľkej podmnožiny údajov, ktorú nie je možné importovať do testovacieho prostredia z dôvodu obmedzenia veľkosti.

7.1.4 Hlásenie a sledovanie chýb

Hlásenie chýb/nedostatkov a následné sledovanie chýb v kódovaní, nedostatkov v návrhu, anomálií v správaní (logických nedostatkov), chýb, porúch a zraniteľností softvéru. Hlásenie chýb by malo byť dostatočne komplexné a podrobné, aby poskytlo tímom pre vývoj softvéru informácie, ktoré sú potrebné na určenie hlavnej príčiny problému.

7.2 Typy testovaní

7.2.1 Post-vývojárske testovanie

Aby sa zabezpečila správnosť vyvinutého softvéru, v rámci tohto kroku sa vykoná analýzu kódu. Ide o dynamickú analýzu kódu, čo je kontrola kódu pri jeho vykonávaní (beží ako program).

7.2.2 Vykonávanie testovania bezpečnosti pomocou metód testovania bezpečnosti

Metódy používané na vykonanie testovania bezpečnosti sú najmä nasledovné:

- tzv. white box testovanie,
- tzv. black box testovanie,
- testovanie kryptografického overenia.

7.2.3 Vykonávanie testovania softvérovej bezpečnosti na zabezpečenie kvality

Predmetom tejto podkapitoly je aplikácia rôznych typov testov a spôsob, akým sa dajú vykonať na overenie bezpečnosti kódu, ktorý sa vyvíja vo fáze vývoja SSDLC.

V prípade revízií softvéru by sa malo vykonať regresné testovanie a pre všetky verzie, nové alebo revízie, by sa mali v prípade potreby vykonať bezpečnostné testy na overenie účinnosti bezpečnostných opatrení. Použitie kategorizovaného zoznamu hrozieb ako šablóny testovania bezpečnosti je účinné pri zabezpečovaní komplexného pokrytia rôznych hrozieb pre softvér. V ideálnom prípade bude zoznam hrozieb, ktorý sa používa aj na vykonávanie bezpečnostných testov, použitý pri modelovaní softvéru.

Medzi najčastejšie vykonávané bezpečnostné testy môžeme zaradiť nasledovné:

- testovanie na overenie vstupov,
- testovanie kontrolných prvkov pre chyby pri zavádzaní,
- testovanie kontrol pre skriptovacie útoky,
- testovanie kontrol spoofingu,
- testovanie kontrol spracovania chýb a výnimiek (testovanie zlyhania),
- testovanie kontrol eskalácie oprávnení,
- záťažové testovanie.

7.3 Začlenenie bezpečnostných testov do celkovej stratégie a plánu testovania

Do finálneho plánu stratégie a testovania musí byť začlenené vykonávanie bezpečnostných testov.

7.4 Vyhodnotenie naplnenia bezpečnostných požiadaviek

Počas fázy testovania je nevyhnutné overiť a vyhodnotiť naplnenie zadaných bezpečnostných požiadaviek, definovaných vo fáze 1 „Požiadavky“. Nenaplnenie jednotlivých požiadaviek by mohlo mať za následok vznik významného bezpečnostného rizika, prípadne neakceptovanie diela a nemožnosť jeho uvedenia do prevádzky.

7.5 Finalizácia bezpečnostného projektu a súvisiacej analýzy rizík

Ide o jednu z finálnych aktivít projektu. V nadväznosti na výsledky realizovaných testov sa zaktualizuje a sfinalizuje bezpečnostný projekt vyvíjaného IS. Rovnako sa zaktualizuje analýza rizík, pričom typicky môžu nastať nasledovné scenáre:

- register rizík zostane nezmenený,
- úroveň jednoho alebo viacerých rizík bude zvýšená,
- úroveň jednoho alebo viacerých rizík bude znížená,
- do registra rizík pribudne nové riziko.

7.6 Roly podieľajúce sa na prevzatí IS do prevádzky

Procesu prevzatia IS do prevádzky sa typicky zúčastňujú nasledovné roly:

- vrcholový zástupca organizácie (budúceho prevádzkovateľa IS),
- zástupca organizácie (budúceho prevádzkovateľa IS) v pozícii kľúčového
- vedúci zamestnanec oddelenia IT prevádzky,
- MKIB.

7.7 Bezpečnostné postupy pri preberaní vyvíjaného IS a nasadzovaní do prevádzky

Ide o bezpečnostné postupy zaisťujúce zachovanie úrovne bezpečnosti IS overenej prostredníctvom bezpečnostného testovania.

Pred ukončením, akceptovaním a odovzdaním projektu je odporúčané zabezpečiť komplexné preverenie bezpečnosti formou vykonania, na dodávateľovi nezávislého, bezpečnostného auditu a najmä penetračného testovania a/alebo „vulnerability scanu“.

Medzi odporúčané bezpečnotné postupy pri nasadzovaní IS do prevádzky by mali patriť aj postupy zaisťujúce zachovanie úrovne bezpečnosti overenej bezpečnostným testovaním. Obsahom týchto postupov by malo byť vytvorenie „čistej“ bezpečnej inštancie vyvinutého IS a príprava bezpečného a spoľahlivého prevádzkového prostredia.

Súčasťou týchto postupov musí byť aj povinné vyhodnotenie a formálne schválenie stavu implementácie bezpečnostných požiadaviek zo strany organizácie. Finálny výstup z tohto vyhodnotenia bude slúžiť ako nevyhnutný podklad ku akceptácii vyvíjaného IS pri jeho preberaní do prostredia organizácie.

7.8 Schválené bezpečnostné konfigurácie

V testovacej fáze je veľmi dôležité, aby sa sfinalizoval register všetkých schválených konfiguračných súborov, ktoré je možné považovať za tzv. konfiguračné šablóny. Je dôležité, aby vývojár organizácii poskytol konfiguračné súbory všetkých aktív, ktoré boli súčasťou projektu; tieto musia následne prejsť schválením MKIB.

7.9 Súlad

Softvér musí byť navrhnutý tak, aby počas jeho budúcej prevádzky boli dodržané príslušné právne predpisy, požiadavky odvetvovej regulácie a technické normy týkajúce sa informačnej bezpečnosti.

7.10 Akceptácia softvéru

Cieľom je zabezpečiť pripravenosť aplikačnej dokumentácie a prevádzkového plánu, získať súhlas a akceptovanie rizika.

7.10.1 Akceptačné kritériá

Je potrebné potvrdiť a overiť všetky funkčné a bezpečnostné požiadavky a ich splnenie voči zadaniu. Kritériá ukončenia pre funkčnosť a bezpečnosť softvéru s explicitnými míľnikmi by mali byť definované v dostatočnom predstihu. Niektoré príklady míľnikov súvisiacich s bezpečnosťou zahŕňajú okrem iného nasledujúce:

- vytvorenie bezpečnostných požiadaviek,
- preskúmanie a podpísanie bezpečnostnej architektúry na konci fázy dizajnu,
- preskúmanie kódu z hľadiska bezpečnostných zraniteľností po ukončení fázy vývoja,
- dokončenie bezpečnostného testovania,
- dokončenie dokumentácie pred začatím fázy nasadenia.

7.10.2 Riadenie zmien

Zmeny prostredia a bezpečnostnej architektúry môžu potenciálne zaviesť nové bezpečnostné zraniteľnosti, čím sa zvyšuje riziko. Organizácia by mala mať zavedené príslušné procesy týkajúce sa riadenia zmien a aplikovať ich aj v rámci SSDLC.

7.10.3 Schválenie nasadenia alebo uvoľnenia

Bez schválenia by nemala byť povolená žiadna zmena v produkčnom prostredí. Pred každou novou inštaláciou softvéru je potrebné vykonať adresnú analýzu rizík a určiť zvyškové riziko. Výsledky analýzy rizík spolu s krokmi prijatými na ich riešenie (zmiernenie alebo akceptovanie) by sa mali oznámiť vlastníkovi organizácie.

Schválenie alebo zamietnutie nasadenia/uvoľnenia by malo obsahovať odporúčania a podporu tímu bezpečnoti.

7.10.4 Politika akceptovania rizika a výnimiek

Zabezpečiť akceptovateľnosť zvyškového rizika a/alebo sledovať ho ako výnimku, ak nie je v rámci prahovej hodnoty. Najprv je potrebné určiť riziko, ktoré zostáva po implementácii bezpečnostných opatrení (reziduálne riziko). Najlepšou možnosťou riešenia celkového rizika je jeho zmiernenie tak, aby zostatkové riziko kleslo pod prahovú hodnotu definovanú organizáciou; v takom prípade je možné zostatkové riziko akceptovať. Riziko by mal akceptovať vlastník rizika.

7.10.5 Dokumentácia softvéru

Zabezpečiť, aby bola k dispozícii všetka potrebná dokumentácia. Zdokumentovanie toho, čo má softvér robiť, ako je navrhnutý, ako sa má nainštalovať, aké konfiguračné nastavenia je potrebné prednastaviť, ako ho používať a spravovať, je mimoriadne dôležité pre efektívne, bezpečné a nepretržité používanie softvéru. Jedným z hlavných cieľov dokumentácie je uľahčiť a zopakovať proces nasadenia softvéru a zabezpečiť, aby nedošlo k narušeniu prevádzky a aby bol správne pochopený vplyv pri zmenách softvéru.

7.10.6 Zachovanie úrovne bezpečnosti

Súčasťou procesov akceptácie IS a jeho nasadenia do prevádzky by mali byť aj postupy zaisťujúce zachovanie úrovne bezpečnosti overenej bezpečnostným testovaním. Obsahom týchto postupov by malo byť vytvorenie „čistej“ bezpečnej inštancie vyvinutého IS a príprava bezpečného a spoľahlivého prevádzkového prostredia.

V postupoch by malo byť zahrnuté predovšetkým:

- vymazanie testovacích údajov,
- zrušenie/zablokovanie účtov vytvorených pre účely vývoja a testovania,
- zmena hesiel pri účtoch, ktoré budú naďalej používané počas prevádzky,
- hardening cieľového prevádzkového prostredia,
- odovzdanie prevádzkovej, administrátorskej a používateľskej dokumentácie,
- formálne akceptovanie finálnych výstupov bezpečnostného projektu,
- formálne akceptovanie opisu splnenia relevantných bezpečnostných štandardov ISVS (môže byť zahrnuté v opise naplnenia bezpečnostných požiadaviek alebo môže tvoriť súčasť bezpečnostného projektu),
- formálne akceptovanie splnenia zmluvne stanovených bezpečnostných požiadaviek,
- integrácia vyvinutého IS s bezpečnostnými nástrojmi v prostredí organizácie (napr. monitorovanie, zálohovanie, riadenie identít),
- bezpečná integrácia na IS, s ktorými sa bude vyvíjaný IS interagovať,
- overenie funkčnosti mechanizmov ochrany kódu pred neoprávnenou zmenou,
- overenie súladu verzie dokumentácie s verziou IS, ktorá je predmetom akceptácie,
- overenie úplnosti dokumentácie konfiguračných nastavení akceptovaného IS.

8 Fáza 6: Prevádzka

SSDLC nekončí akceptáciou vyvíjaného softvéru, ale pokračuje fázou prevádzky.

V rámci tejto fázy je z pohľadu bezpečnosti dôležité najmä formálne riadenie zmien, ich posudzovanie, analýza a schvaľovanie.

Pri procese formálneho riadenia zmien organizácia postupuje v zmysle vlastných interných postupov týkajúcich sa riadenia zmien; súčasťou tejto kapitoly je príklad osvedčenej praxe.

8.1 Riadenie nasadzovania do prevádzky

Zabezpečenie správneho uvoľnenia softvéru do prevádzkového prostredia po tom ako sú hardvérové a softvérové prostriedky a prostredie nakonfigurované na bezpečnú prevádzku.

Riadenie nasadzovania do prevádzky je proces zabezpečujúci, aby všetky zmeny, ktoré sa vykonajú v produkčnom prostredí boli naplánované, zdokumentované, dôkladne otestované a nasadené bez toho, aby negatívne ovplyvnili akékoľvek existujúce obchodné operácie, zákazníkov, koncových používateľov alebo tímy podpory používateľov.

8.2 Zaznamenanie a kategorizácia požiadavky na zmenu

Požiadavku na zmenu môže iniciovať akýkoľvek konečný používateľ pri zistení nedostatku, pri zistení príslušnej zmeny právnych predpisov, pri zistení nedostatočnej funkcionality aplikácie atď. Všetky požiadavky na zmenu musia byť zaznamenané.

Pri kategorizácii požiadaviek na zmenu organizácia postupuje v zmysle vlastných interných postupov týkajúcich sa riadenia zmien; typicky je však možné požiadavky zatriediť do niektorej z nasledovných kategórií:

- požiadavka na zmenu týkajúca sa softvéru,
- požiadavka na zmenu týkajúca sa hardvéru,
- požiadavka na zmenu týkajúca sa služby.

8.3 Schvaľovanie a klasifikácia požiadavky na zmenu

Po identifikácii a zaznamenaní požiadavky na zmenu pracovník v roli schvaľovateľ zmeny overí, či je požiadavka na zmenu zrozumiteľná, jasná, či nie je nelogická alebo nepotrebná.

V prípade, že je požiadavka na zmenu zamietnutá, žiadateľ je o jej zamietnutí informovaný aj s uvedením dôvodu jej zamietnutia. Ak je požiadavka na zmenu akceptovaná, je ďalej klasifikovaná, prioritizovaná a je stanovený tzv. vlastník zmeny.

Pod pojmom klasifikácia požiadavky je možné chápať určenie priority požiadavky a jej dopadu na IT organizáciu.

Priorita vyjadruje dôležitosť žiadanej zmeny v porovnaní s ďalšími požiadavkami na zmenu a vyjadruje taktiež naliehavosť a potrebu implementácie zmeny zo strany konečného používateľa. Priorita môže byť stanovená žiadateľom o zmenu vo formulári, avšak táto môže byť prehodnotená v kontexte celej množiny zmien v organizácii. Priorita je stanovovaná z dôvodu, aby naliehavejšie zmeny boli realizované skôr ako zmeny menej naliehavé. Z pohľadu priority sa odporúča prioritizovať zmeny

najmenej na zmeny s:

- nízkou prioritou – zmena je potrebná ale jej uskutočnenie nie je potrebné okamžite,
- strednou prioritou – zmena je naliehavejšia a má relatívne vysoký dopad na IT organizáciu.
- vysokou prioritou – vysoká priorita je stanovená v prípade, ak ide o požiadavku o naliehavú zmenu, ktorej prípadné nevyriešenie v krátkom čase by malo za následok pozastavenie služieb na dlhšie časové obdobie s vplyvom na veľa konečných používateľov

Z pohľadu dopadu zmeny na IT organizáciu sa odporúča nasledovný spôsob klasifikácie:

- minimálny dopad – zmena, ktorá vyžaduje menej práce s nízkym rizikom,
- značný dopad – zmena, ktorá vyžaduje značnú snahu a bude mať značný dopad na poskytované služby,
- významný dopad – zmena, ktorá vyžaduje veľké úsilie a môže ovplyvniť značnú časť IT organizácie.

V prípade, že má požadovaná zmena vplyv na kybernetickú bezpečnosť, jedným zo schvaľovateľov musí byť MKIB.

8.4 Plánovanie implementácie požiadavky na zmenu

Po stanovení priority a dopadu schválených požiadaviek na zmenu je potrebné detailnejšie naplánovanie ich realizácie v závislosti od miery vplyvu na informačnú bezpečnosť, nákladov, zdrojov, času potrebného na implementáciu zmeny, vplyvu na prevádzku a pod.

Požiadavky na zmenu môžu byť typicky riešené interne alebo dodávateľským spôsobom. V prípade, že bude realizácia zmeny vykonávaná dodávateľským spôsobom, je potrebné, aby dodávateľom odhadnuté požadované zdroje (aj s odhadom nákladov – personálnych a finančných) boli schválené na príslušnej úrovni organizácie.

8.5 Návrh riešenia zmeny

Akceptované zmeny sú ďalej postúpené príslušnému špecialistovi, ktorý bude riešiť jej realizáciu.

Všetky zmeny nemusia obsahovať detailný návrh riešenia; detailný návrh riešenia nie je potrebný najmä pri tzv. štandardných zmenách, ktoré môžu byť riešené skráteným procesom riadenia zmien.

Návrh riešenia môže zahŕňať napr. implementáciu novej verzie softvéru, spolu s vytvorením potrebných manuálov, dokumentácie, príručiek (technická, používateľská atď.) a záložného plánu alebo zmeny hardvéru. Vo fáze návrhu riešenia môžu byť zahrnuté rôzne podprocesy ako napr. dôkladná analýza požiadaviek, verejné obstarávanie a pod.

V prípade, že má zmena dopad na kybernetickú bezpečnosť, do návrhu riešenia musí byť zapojený MKIB a útvar bezpečnosti, aby bolo zabezpečené, že všetky bezpečnostné požiadavky budú v návrhu zachytené.

V rámci fázy návrhu riešenia musia byť navrhnuté testovacie scenáre v súlade s požiadavkami uvedenými v požiadavke na zmenu.

Návrh riešenia zmeny schvaľuje schvaľovateľ zmeny. Po schválení návrhu je žiadateľ o zmenu informovaný o predpokladanom termíne implementácie zmeny.

8.6 Testovanie zmeny

Pred samotnou implementáciou zmeny do produkčného prostredia musí prebehnúť riadne otestovanie zmeny v testovacom prostredí, aby sa vylúčili neúmyselné vplyvy na existujúce systémy a aplikácie. Testovanie musí prebiehať v testovacom prostredí, ktoré je oddelené od produkčného prostredia. Testovanie musí byť vykonané v súlade s požiadavkou na zmenu a testovacími scenármi vytvorenými vo fáze návrhu riešenia.

Testovanie musí byť vhodným a preukázateľným spôsobom zdokumentované. Na testovaní sa podieľajú – v závislosti od typov testovacích scenárov – žiadateľ o zmenu, vlastník zmeny, implementátor, MKIB, zástupca oddelenia IT prevádzky, prípadne ďaší zástupcovia organizácie.

V praxi sa realizujú v princípe dva druhy testov:

- testy funkcionality,
- akceptačné používateľské testy.

8.7 Implementácia zmeny

Požadovaná zmena môže byť nasadená do produkcie až po úplnej akceptácii riešenia všetkými relevantnými pracovníkmi. Finálne riešenie schvaľuje žiadateľ zmeny, kľúčový používateľ, zástupca IT prevádzky a v prípade požiadavky s dopadom na kybernetickú bezpečnosť i MKIB.

Po samotnej implementácii sa vytvorí a parafuje akceptačný protokol, ktorý obsahuje popis testovaných oblastí, výsledok testovania a samotnú akceptáciu.

8.8 Vyhodnotenie a uzatvorenie požiadavky na zmenu

Vyhodnotenie a uzatvorenie požiadavky na menu je zodpovednosťou vlastníka zmeny.

Vlastník zmeny po jej implementácii iniciuje jej post-implementačné hodnotenie. Samotné post-implementačné hodnotenie vykonáva žiadateľ o zmenu a obsahuje odpovede na otázky, či bol realizáciou zmeny dosiahnutý požadovaný výsledok, spokojnosť konečných používateľov, informáciu o dodržaní/presiahnutí predpokladaných nákladov, informáciu o informovaní dotknutých strán a pod.

V prípade úspešnej implementácie zmeny môže byť požiadavka na zmenu uzatvorená a vyhodnotenie zaznamenané v samotnej požiadavke na zmenu. V prípade problémov počas implementácie zmeny je potrebné uviesť problémy, ktoré sa počas implementácie vyskytli.

Neúspešne implementované zmeny musia byť bezodkladne riešené vedením organizácie.

Všetky implementované zmeny by mali byť naďalej vyhodnocované na pravidelnej báze.

9 Fáza 7: Vyradenie

Pri vyradení systému je potrebné zabezpečiť archiváciu údajov alebo ich nenávratné zmazanie (ak nemajú trvalú dokumentárnu hodnotu), vyčistenie alebo fyzickú likvidáciu záznamových médií, prípadne upraviť ďalšie disponovanie s hardvérom a softvérom.

9.1 Archivácia údajov

Pri archivácii údajov je nutné vytvoriť špecifickú politiku archivácie, ktorá by sa zaoberala dlhodobým uchovávaním údajov s trvalou dokumentárnou hodnotou.

V súvislosti s archiváciou údajov je potrebné zabezpečiť najmä:

- určenie okruhu údajov, ktoré budú predmetom archivácie a stanovenie doby ich archivácie (v súlade s registratúrnym poriadkom organizácie),
- definovanie požiadaviek na bezpečnosť archívnych údajov a vytvorenie pravidiel pre manipuláciu s archivačnými médiami,
- zabezpečenie vhodných podmienok pre uchovávanie archivačných médií,
- vytvorenie presných a úplných záznamov o archivačných médiách,
- pravidelné riadené testovanie archivačných médií,
- dlhodobé úložisko dokumentov a pravidlá pre presun archivovaných údajov do štátneho archívu,
- a pod.

9.2 Likvidácia údajov

Citlivé informácie by sa nemali uchovávať dlhšie, ako je potrebné na zníženie rizika nežiaducich udalostí. Pri odstraňovaní informácií o systémoch, aplikáciách a službách je potrebné zvážiť nasledovné:

- výber spôsobu vymazania (napr. elektronické prepisovanie alebo kryptografické vymazanie) v súlade s obchodnými požiadavkami a so zohľadnením príslušných zákonov a nariadení,
- zaznamenanie výsledkov vymazania ako dôkazu,
- pri využívaní poskytovateľov služieb na vymazanie informácií, získanie dôkazov o vymazaní informácií.

Ak tretie strany ukladajú informácie organizácie v jej mene, organizácia by mala zvážiť zahrnutie požiadaviek na vymazanie informácií do dohôd s tretími stranami s cieľom ich presadzovania počas a po ukončení poskytovania takýchto služieb.

V súlade s tematickou politikou organizácie týkajúcou sa uchovávania údajov a s ohľadom a prihliadnutím na príslušné právne predpisy a nariadenia by sa citlivé informácie mali vymazať, keď už nie sú potrebné, a to:

- konfigurácia systémov na bezpečné zničenie informácií, keď už nie sú potrebné (napr. obdobie podliehajúce politike uchovávania údajov alebo žiadosti o prístup subjektu),
- mazanie zastaraných verzií, kópií a dočasných súborov bez ohľadu na to, kde sa nachádzajú,
- používanie schváleného softvéru na bezpečné vymazanie na trvalé vymazanie informácií,

aby sa zabezpečilo, že informácie nie je možné obnoviť pomocou špecializovaných nástrojov na obnovu alebo forenzných nástrojov.

Ak sa využívajú cloudové služby, organizácia by mala overiť, či metóda vymazávania poskytovaná poskytovateľom cloudových služieb, je prijateľná, a ak je to tak, organizácia by ju mala používať alebo požiadať, aby poskytovateľ cloudových služieb vymazal informácie. Tieto procesy vymazávania by mali byť automatizované a v súlade s politikami špecifickými pre danú tému, ak sú k dispozícii a použiteľné. V závislosti od citlivosti vymazaných informácií možno v protokoloch sledovať alebo overovať, či sa tieto procesy vymazania uskutočnili.

Pri likvidácii údajov je možné využiť i poskytovateľov služieb bezpečnej likvidácie hardvérových komponentov. Pri tomto type likvidácie údajov je potrebné vytypovať likvidačné mechanizmy vhodné pre konkrétny typ hardvéru, ktoré sa likviduje (napr. demagnetizácia pevných diskov a iných magnetických pamäťových médií, fyzické zošrotovanie IT komponentov a pod.).

Ďalšie disponovanie s hardvérom, na ktorom boli v minulosti uložené vymazané údaje musí byť schválené zo strany MKIB; v opačnom prípade nie je možné takýto hardvér použiť na iné účely.

10 Prílohy

10.1 Príloha 1 – Konkrétne príklady bezpečnostných požiadaviek

10.1.1 Príklady bezpečnostných požiadaviek podľa ISO/IEC 15408

Common Criteria for Information Technology Security Evaluation (ďalej aj „Common Criteria“ alebo „CC“) je medzinárodná norma ISO/IEC 15408 pre certifikáciu počítačovej bezpečnosti. Hodnotenia CC sa vykonávajú na produktoch a systémoch počítačovej bezpečnosti.

Cieľ hodnotenia (ďalej aj „TOE“) predstavuje produkt alebo systém, ktorý je predmetom hodnotenia. Hodnotenie slúži na overenie tvrdení o cieli hodnotenia. Aby malo hodnotenie praktický význam, musí overiť bezpečnostné funkcie cieľa; toto sa uskutočňuje prostredníctvom nasledujúcich krokov:

- Profil ochrany (ďalej aj „Protection Profile“ alebo „PP“) je dokument, ktorý zvyčajne vytvorí používateľ alebo komunita používateľov a ktorý identifikuje bezpečnostné požiadavky na triedu bezpečnostných zariadení (napríklad čipové karty používané na zabezpečenie digitálnych podpisov alebo sieťové firewally) relevantné pre daného používateľa na konkrétny účel. Dodávatelia produktov sa môžu rozhodnúť implementovať produkty, ktoré sú v súlade s jedným alebo viacerými PP, a nechať svoje produkty vyhodnotiť na základe týchto PP. V takom prípade môže PP slúžiť ako vzor pre Security Target (ďalej aj „ST“) produktu (bezpečnostný cieľ, ako je definovaný ďalej) alebo autori ST aspoň zabezpečia, aby sa všetky požiadavky v príslušných PP objavili aj v dokumente ST cieľa. Zákazníci, ktorí hľadajú konkrétne typy produktov, sa môžu zamerať na produkty certifikované podľa PP, ktoré spĺňajú ich požiadavky.
- Security Target (ST) – dokument, ktorý identifikuje bezpečnostné vlastnosti cieľa hodnotenia. ST môže požadovať zhodu s jedným alebo viacerými PP. TOE sa hodnotí podľa bezpečnostných funkčných požiadaviek (ďalej aj „SFR“) ustanovené v jeho ST. To umožňuje predajcom prispôsobiť hodnotenie tak, aby presne zodpovedalo zamýšľaným schopnostiam ich produktu. To znamená, že sieťový firewall nemusí spĺňať rovnaké funkčné požiadavky ako systém správy databáz a že rôzne firewally môžu byť v skutočnosti hodnotené podľa úplne odlišných zoznamov požiadaviek. ST sa zvyčajne zverejňuje, aby potenciálni zákazníci mohli určiť špecifické bezpečnostné prvky, ktoré boli certifikované hodnotením.
- Bezpečnostné funkčné požiadavky (SFR) – špecifikujú jednotlivé bezpečnostné funkcie, ktoré môže produkt poskytovať. Common Criteria predstavuje štandardný katalóg takýchto funkcií. Napríklad SFR môže uvádzať, ako môže byť autentifikovaný používateľ s určitou rolou. Zoznam SFR sa môže líšiť od jedného hodnotenia k druhému, aj keď dva ciele predstavujú rovnaký typ produktu. Hoci Common Criteria nepredpisuje žiadne SFR, ktoré majú byť zahrnuté v ST, identifikuje závislosti, kde správna činnosť jednej funkcie (napríklad schopnosť obmedziť prístup podľa rolí) závisí od inej (napríklad schopnosť identifikovať jednotlivé roly).

10.1.2 Príklady bezpečnostných požiadaviek podľa OWASP

Dôvernosť prenosu

Dôvernosť prenosu chráni pred odpočúvaním a útokmi typu man-in-the-middle proti komunikácii webových služieb do/zo servera.

Všetka komunikácia s webovými službami a medzi nimi, ktoré obsahujú citlivé funkcie, overená relácia alebo prenos citlivých údajov, musí byť šifrovaná pomocou dobre nakonfigurovaného TLS. Odporúča

sa to aj vtedy, keď sú samotné správy šifrované, pretože TLS poskytuje množstvo výhod nad rámec dôvernosti prenosu vrátane ochrany integrity, ochrany proti opakovanému prehrávaniu a autentifikácie servera.

Autentifikácia servera

TLS sa musí použiť na autentifikáciu poskytovateľa služby pre spotrebiteľa služby.

Zákazník služby by si mal overiť, že certifikát servera vydal dôveryhodný poskytovateľ, nevypršala platnosť, nie je odvolaná, zhoduje sa s názvom domény služby a že server preukázal, že má súkromný kľúč spojený s certifikátom verejného kľúča (riadnym podpísaním niečoho alebo úspešným dešifrovaním niečoho zašifrovaného pomocou priradeného verejného kľúča).

Autentifikácia používateľa

Autentifikácia používateľa overuje identitu používateľa alebo systému, ktorý sa pokúša pripojiť k službe. Takáto autentifikácia je zvyčajne funkciou kontajnera webovej služby.

Autentifikácia klientskeho certifikátu pomocou Mutual-TLS je bežná forma autentifikácie, ktorá je odporúčaná.

Kódovanie prenosu

Štýly kódovania SOAP sú určené na presun údajov medzi softvérovými objektmi do formátu XML a späť. Pri komunikácii je nutné vynútiť rovnaký štýl kódovania medzi klientom a serverom.

Integrita správy

Integritu prenášaných údajov možno jednoducho zabezpečiť pomocou TLS.

Pri použití kryptografie s verejným kľúčom šifrovanie zaručuje dôvernosť, ale nezaručuje integritu, pretože verejný kľúč príjemcu je verejný. Z rovnakého dôvodu šifrovanie nezabezpečuje identitu odosielateľa.

Pre údaje XML je možné použite digitálnych podpisov XML na zabezpečenie integrity správy pomocou súkromného kľúča odosielateľa. Tento podpis môže príjemca overiť pomocou digitálneho certifikátu odosielateľa (verejného kľúča).

Dôvernosť správy

Dátové prvky, ktoré majú byť dôverné, musia byť zašifrované pomocou silnej šifrovacej šifry s primeranou dĺžkou kľúča, aby sa zabránilo hrubému vynúteniu.

Správy obsahujúce citlivé údaje musia byť šifrované pomocou silného šifrovania. Môže to byť prenosové šifrovanie alebo šifrovanie správ.

Správy obsahujúce citlivé údaje, ktoré musia po prijatí zostať v pokoji zašifrované, musia byť zašifrované silným šifrovaním údajov, nielen prenosovým šifrovaním.

Autorizácia

Webové služby musia autorizovať klientov webových služieb rovnakým spôsobom, akým webové aplikácie autorizujú používateľov. Webová služba sa musí uistiť, že klient webovej služby je oprávnený vykonať určitú akciu s požadovanými údajmi.

Webová služba by mala autorizovať svojich klientov, či majú prístup k danej metóde. Po výzve na overenie by webová služba mala skontrolovať oprávnenia žiadajúcej entity, či má prístup k požadovanému zdroju. Takéto overenie by sa malo vykonať pri každej žiadosti. K citlivým zdrojom, ako sú zmeny hesla, primárne kontaktné údaje, ako je e-mail, fyzická adresa, pokyny na platbu alebo doručenie by sa mal pridať mechanizmus autorizácie typu výzva-odpoveď.

Prístup k funkciám správy a správy v rámci aplikácie webovej služby by mal byť obmedzený na správcov webových služieb. V ideálnom prípade by všetky administratívne funkcie mali byť v aplikácii, ktorá je úplne oddelená od webových služieb spravovaných týmito schopnosťami, čím by boli normálni používatelia úplne oddelení od týchto citlivých funkcií.

Overenie schémy

Webové služby musia overiť užitočné zaťaženia SOAP podľa ich pridruženej definície schémy XML (XSD). XSD definovaný pre webovú službu SOAP by mal minimálne definovať maximálnu dĺžku a znakovú sadu každého parametra, ktorý môže prejsť do webovej služby a von z nej.

XSD definovaný pre webovú službu SOAP by mal definovať silné overovacie vzory pre všetky parametre pevného formátu (napr. PSČ, telefónne čísla, hodnoty zoznamu atď.).

Overenie obsahu

Ako každá webová aplikácia, aj webové služby musia pred použitím overiť vstup. Overenie obsahu pre vstup XML by malo zahŕňať:

- overenie proti chybným entitám XML,
- overenie proti útokom XML Bomb,
- overenie vstupov pomocou silného zoznamu povolených,
- overenie proti útokom vonkajších entít.

Kódovanie výstupu

Webové služby musia zabezpečiť, aby výstup odosielaný klientom bol zakódovaný tak, aby sa spotreboval ako údaje a nie ako skripty. Toto je dosť dôležité, keď klienti webových služieb používajú výstup na vykresľovanie stránok HTML priamo alebo nepriamo pomocou objektov AJAX.

Ochrana pred vírusmi

SOAP poskytuje možnosť pripojiť súbory a dokumenty k správam SOAP. To dáva hackerom príležitosť pripojiť k týmto SOAP správam vírusy a malvér.

Zabezpečte, aby bola technológia Virus Scanning pravidelne aktualizovaná najnovšími definíciami/pravidlami vírusov.

Veľkosť správy

Webové služby, ako sú webové aplikácie, by mohli byť cieľom útokov DOS automatickým odosielaním tisícok veľkých správ SOAP webovým službám. To buď ochromí aplikáciu, takže nebude môcť odpovedať na legitímne správy alebo ju môže úplne zrušiť.

Veľkosť SOAP správ by mala byť obmedzená na vhodný limit veľkosti. Väčší limit veľkosti (alebo žiadny limit) zvyšuje šance na úspešný DoS útok.

Obmedzenie zdrojov

Počas bežnej prevádzky webové služby vyžadujú výpočtový výkon, ako sú cykly CPU a pamäť. V dôsledku nefunkčnosti alebo útoku môže webová služba vyžadovať príliš veľa zdrojov, čo spôsobí, že hostiteľský systém bude nestabilný.

Obmedzte množstvo cyklov CPU, ktoré môže webová služba použiť na základe očakávanej rýchlosti služieb, aby bol systém stabilný.

Obmedzte množstvo pamäte, ktorú môže webová služba použiť, aby ste zabránili nedostatku pamäte systému. V niektorých prípadoch môže hostiteľský systém začať zabíjať procesy, aby uvoľnil pamäť.

Obmedzte počet súčasne otvorených súborov, sieťových pripojení a spustených procesov.

Priepustnosť správ

Priepustnosť predstavuje počet žiadostí o webovú službu obsluhovaných počas určitého času.

Konfigurácia by mala byť optimalizovaná pre maximálnu priepustnosť správ, aby ste sa vyhli situáciám podobným DoS.

Ochrana pred XML Denial of Service

XML Denial of Service je pravdepodobne najvážnejším útokom proti webovým službám. Takže webová služba musí poskytnúť nasledujúce overenie:

- validácia proti rekurzívnemu užitočnému zaťaženiu,
- overenie proti nadmernému užitočnému zaťaženiu,
- ochrana pred expanziou XML entity,
- overenie proti príliš dlhým názvom prvkov. Ak pracujete s webovými službami založenými na SOAP, názvy prvkov sú tieto akcie SOAP.

Túto ochranu by mal poskytovať váš analyzátor XML/schéma validátor. Na overenie vytvorte testovacie prípady, aby ste sa uistili, že váš syntaktický analyzátor je odolný voči týmto typom útokov.