Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Aplikačná bezpečnosť
Časť 2

Erik Saller, Ivan Oravec

KPMG
cutting through complexity™

Osnova

- Úvod
- Základné bezpečnostné hrozby pre aplikácie
- Aplikačné bezpečnostné funkcie
- Typické zraniteľnosti aplikácií a opatrenia proti nim
- Používanie otvorených štandardov
- Záver

Vkladanie neoprávnených SQL dotazov (SQL Injection)

- Útoky „SQL injection“ sa realizujú vkladaním, alebo tiež injekciou SQL kódu do vstupných dát prenášaných medzi klientom a aplikáciou.
- Útoky SQL injekcie sú takými typmi injekčných útokov, v ktorých sú príkazy SQL injektované do existujúcich legitímnych SQL vstupov tak, aby došlo k spusteniu útočníkom preddefinovaných modifikovaných SQL príkazov.

Vkladanie neoprávnených SQL dotazov (SQL Injection) – pokr.

- Využitie SQL injekcie, ktorá úspešne prekoná bezpečnostné funkcie aplikácie je umožňuje útočníkovi:
  - čítať citlivé dáta z databázy,
  - modifikovať dáta v databáze (vkladanie, aktualizácia, mazanie),
  - spúšťať administratívne operácie nad databázou (vypnutie systému riadenia databázy – SRBD, anglicky DBMS),
  - obnovenie obsahu určitého súboru prítomného v databáze a
  - v špeciálnych prípadoch tiež spúšťanie príkazov na operačnom systéme.

Hrozby

- útoky SQL injekcie umožňujú útočníkovi:
  - využívať cudziu identitu,
  - modifikovať existujúce dáta,
  - spôsobovať problémy pri prevádzaní transakcií (zrušenie transakcie, zmenenie kľúčových hodnôt pri peňažných transakciách ako je napr. výška prevádzanej sumy),
  - úplne prezradenie všetkých citlivých informácií v systéme,
  - zničenie dát, alebo ich premiestnenie do nežiadanej lokality a zamedzenie prístupu k nim pre bežných užívateľov a
  - získanie administrátorských práv k napadnutému systému,

Hrozby

- útok typu SQL injection je veľmi bežný pri PHP a ASP aplikáciách kvôli tomu, že sa medzi nimi stále bežne používajú staršie funkčné rozhrania.
- Kvôli charakteru prístupných rozhraní, sú J2EE a ASP.NET aplikácie menej zraniteľné na bežné útoky typu SQL injection.
- Podobne ako pri iných typoch útokov (ako napr. XSS), je závažnosť útokov typu SQL injection výrazne ovplyvnená schopnosťami a kreativitou útočníka.

Hrozby (pokr.)

- Rolu tu hrajú tiež protiopatrenia na strane servera, ako je napríklad nastavenie nízkych privilégií pre spojenia s databázovým serverom atď.
- Vo všeobecnosti je potrebné považovať útoky typu SQL injekcie za vysoko závažné.

Hrozby (pokr.)

- K útokom formou SQL injekcie dochádza keď:
  - dáta vstupujú do aplikácie z nedôveryhodného zdroja,
  - dáta, ktoré vstupujú do aplikácie z dôveryhodného zdroja si dynamicky vytvárajú SQL požiadavku.
- Útoky vkladaním SQL kódu sa stali bežným problémom pri webových aplikáciách používajúcich databázu. Chyba je často ľahko detekovateľná a zneužiteľná a preto je ktorákoľvek webová stránka, alebo softvérový balíček s aspoň minimálnym vstupom od užívateľa ľahko kompromitovaná týmto typom útoku.

Hrozby (pokr.)

- V zásade je útok založený na modifikácii existujúceho dátového vstupu vložením meta znaku, za ktorým nasleduje nový, útočníkom vykonštruovaný SQL príkaz, ktorý v ňom predtým neexistoval. Táto chyba zneužíva vlastnosť aplikácie netestovať a nerozlišovať vzory vo vstupných dátach.
- Tradičnou metódou na prevenciu útokov SQL injekcie je pristupovať k nim ako ku problému s dátovou validáciou a buď akceptovať iba znaky z určitého vopred definovaného zoznamu (z tzv. „whitelist“-u) bezpečných hodnôt, alebo identifikovať a vylúčiť potenciálne nebezpečné hodnoty z vopred definovaného zoznamu (tzv. „blacklist“-u).

Cross-site Scripting (XSS)

- Cross-Site Scripting útoky sú takým typom injekčného útoku, v ktorom sú potenciálne škodlivé skripty injektované do inak neškodných a dôveryhodných webových stránok.
- Cross-Site Scripting (XSS) útoky nastávajú, keď útočník použije webovú aplikáciu na zaslanie škodlivého kódu inému používateľovi.

Cross-site Scripting (XSS) – pokr.

- Vo väčšine prípadov sa jedná o škodlivý kód vo forme skriptu na strane prehliadača.
- Chyby, ktoré umožňujú úspešnosť takéhoto útoku sú vo všeobecnosti pomerne rozšírené a nastávajú kdekoľvek, kde webová aplikácia využíva vstup od používateľa na generovanie vlastného výstupu bez toho, aby ho overovala, alebo zakódovala.

Cross-site Scripting (XSS) (pokr.)

- Útočník môže použiť XSS na zaslanie škodlivého skriptu nič netušiacemu používateľovi.
- Prehliadač koncového používateľa nemá možnosť zistiť, že tento skript nie je dôveryhodný a spustí ho.

Cross-site Scripting (XSS) (pokr.)

- Pretože si prehliadač myslí, že skript pochádza od dôveryhodného zdroja, skript môže pristupovať k:
  - ľubovoľným súborom „cookies“,
  - identifikátorom aktívnej session, alebo
  - iným citlivým informáciám udržiavaným prehliadačom používateľa a použitým touto stránkou.
  - Tieto skripty môžu dokonca prepísať obsah HTML stránky.

Cross-site Scripting (XSS) (pokr.)

- Cross-Site Scripting (XSS) útoky nastávajú, keď:
  - dáta vstupujú do webovej aplikácie cez nedôveryhodný zdroj, najčastejšie formou webovej požiadavky (ide o bežné HTTP volanie, typicky to býva napríklad kliknutie na vybraný objekt web stránky a následné zavolanie konkrétneho URL s ktorým sú zasielané aj určité vstupné parametre)
  - dáta sú zahrnuté v dynamickom obsahu, ktorý je používateľovi webovej stránky zaslaný bez toho, aby bol validovaný na prítomnosť škodlivého kódu.

Cross-site Scripting (XSS) (pokr.)

- Škodlivý obsah zaslaný webovému prehliadaču má zväčša formu:
  - časti JavaScriptového kódu, ale
  - môže tiež obsahovať kód HTML,
  - Flashový kód, alebo
  - iný typ kódu, ktorý je prehliadač schopný spustiť.

Cross-site Scripting (XSS) (pokr.)

- Rôznorodosť útokov založených na XSS je takmer neobmedzená, ale vo väčšine prípadov zahŕňa nelegitímne zaslanie súkromných dát, akými sú:
  - cookies, alebo
  - iné informácie o aktívnej session smerom k útočníkovi,
  - presmerovanie obete na webový obsah pod kontrolou útočníka, alebo
  - vykonanie inej škodlivej operácie na počítači obete prostredníctvom zraniteľnej webovej stránky.

  „Zraniteľná webová stránka“ ešte neznamená, že ju útočník ovláda.

Uložené a reflektované XSS útoky

- Cross-Site Scripting útoky môžu byť vo všeobecnosti zatriedené do dvoch kategórií: uchovávané (stored) a reflektované (reflected). Existuje tiež tretia, menej známa kategória zvaná „DOM based XSS“.

Uložené XSS útoky

- Uložené útoky sú také, pri ktorých je injektovaný kód permanentne uchovávaný na cieľových serveroch, napríklad v databáze, v diskusnom fóre, návštevnej knihe, v políčku pre komentár, atď. Obeť príjme škodlivý skript zo servera pri požiadavke o uloženú informáciu.

Zrkadlené XSS útoky

- Reflektované útoky sú také, pri ktorých je injektovaný kód „zrkadlený“ z webového servera.
- Napríklad sa môže jednať o chybové hlásenie, výsledok vyhľadávania, alebo inú odpoveď, ktorá zahŕňa časť vstupu, alebo celý vstup zasielaný serveru ako časť požiadavky. Reflektované útoky sú smerované na obete útoku cez iný kanál, napríklad emailovou správou, alebo cez odlišný webový server.

Zrkadlené XSS útoky

- Keď je užívateľ navedený na kliknutie na linku, alebo zaslanie špeciálne modifikovaného formulára, za ktorým sa skrýva škodlivý kód, injektovaný kód putuje do zraniteľného webového servera, ktorý „zrkadlí“ útok späť k používateľskému prehliadaču.
- Prehliadač potom spúšťa škodlivý kód preto, že tento kód pochádza z „dôveryhodného“ zdroja.

DOM Based XSS – útoky založené na modifikácii DOM

- Útok nazývaný DOM Based XSS (iné používané meno je „type-0 XSS“) je taký XSS útok, v ktorom sú škodlivé dáta prenášané pri útoku spustené ako výsledok modifikácie DOM prostredia („Document Object Model“) v prehliadači používateľa používanom originálnym skriptom na strane klienta tak, že kód na strane klienta beží v „neočakávanom“ režime.

DOM Based XSS – útoky založené na modifikácii DOM (pokr.)

- To znamená, že samotná stránka (teda http odozva na požiadavku klienta) sa nezmení, ale kód na strane klienta obsiahnutý v stránke sa spúšťa rozdielne kvôli škodlivej modifikácii útočníka, ktorá sa udiala v DOM prostredí.
- Tento útok je odlišný od ostatných XSS útokov (uložených, alebo reflektovaných), pretože škodlivé dáta sú uložené v stránke, ktorá je odpoveďou na požiadavku (vďaka chybe na strane servera).

Pretečenie zásobníka (pamäte)

- Chyba typu pretečenie zásobníka nastáva, keď aplikácia pri zapisovaní údajov do rôznych dátových štruktúr nekontroluje, prípadne kontroluje nedostatočne, či veľkosť zapisovaných údajov nie je väčšia než veľkosť dátovej štruktúry do ktorej sa zápis vykonáva.
- Ak je veľkosť zapisovaných údajov väčšia, po vyčerpaní priestoru v alokovanej dátovej štruktúre môže dôjsť k prepísaniu rôznych aplikačných alebo systémových riadiacich dátových štruktúr. V takomto prípade najčastejšie dochádza k havárii aplikácie a ukončeniu jej činnosti.

Pretečenie zásobníka (pamäte)

- Ak však útočník dokáže ovplyvňovať (dodať) údaje, ktorými bude tento prepis vykonaný, môže okrem zhodenia aplikácie dosiahnuť aj spustenie vlastného kódu na cieľovom systéme.
- Kód, ktorý chce útočník spustiť obvykle posiela aplikácii v rámci údajov, ktorými bude vykonaný prepis spomínaných dátových štruktúr.
- Útočníci vo všeobecnosti používajú pretečenia zásobníka pre narušenie priebehu spúšťania kódu aplikácie.

Pretečenie zásobníka (pamäte)

- Pomocou vkladania modifikovaných vstupov do aplikácie je útočník schopný spustiť cudzí kód, ktorý môže prebrať kontrolu nad systémom, na ktorom aplikácia beží.
- Historicky boli identifikované chyby zneužiteľné na útok pretečenia zásobníka vo veľkom množstve produktov a komponentov.

Pretečenie zásobníka (pamäte) (pokr.)

- Chyby zneužiteľné na pretečenie zásobníka môžu byť prítomné v produktoch určených na prevádzkovanie webových služieb aj v produktoch určených pre poskytovanie ostatných aplikačných služieb, ktoré prevádzkujú statickú, alebo dynamickú časť stránky, ale tiež môže byť prítomná v samotnej webovej aplikácii.
- Chyby zneužiteľné na útok pretečením zásobníka nájdené v bežne používaných produktoch sa s veľkou pravdepodobnosťou stávajú široko známymi a predstavujú významné riziko pre používateľov týchto produktov.

Pretečenie zásobníka (pamäte) (pokr.)

- Pokiaľ napr. webová aplikácia používa knižnice, ako napríklad grafické knižnice na generovanie obrázkov, alebo komunikačné knižnice na posielanie emailov, vystavuje sa tým potenciálnym útokom pretečením zásobníka.
- Literatúra, ktorá popisuje útoky pretečením zásobníka voči bežne používaným produktom je voľne dostupná a nové zraniteľnosti sú publikované takmer denne.

Pretečenie zásobníka (pamäte) (pokr.)

- Chyby pretečenia zásobníka sa dajú nájsť tiež v upravenom kóde proprietárnych aplikácií a môžu byť zneužité dokonca s väčšou pravdepodobnosťou, pretože neprešli podrobnejším skúmaním a testovaním, ktorým bežne používané aplikácie väčšinou prechádzajú.
- Útoky pretečením zásobníka na aplikácie často vedú k prekvapivým výsledkom.

Pretečenie zásobníka (pamäte) (pokr.)

- V niektorých prípadoch má vloženie veľkého množstva vstupu za následok zlyhanie aplikácie, alebo databázy na pozadí aplikácie.
- V závislosti na závažnosti a charaktere chyby je dokonca možné spôsobiť útok odopretia služby.
- Veľké množstvo dát posunuté rozhraniu aplikácie môže aplikáciu prinútiť k zobrazeniu popisného chybového hlásenia, ktoré môže potenciálne viesť k úspešnému útoku na systém.

Pretečenie zásobníka (pamäte) (pokr.)

- Útoky pretečenia zásobníka sa vo všeobecnosti realizujú dvoma technikami, pričom sa často používa ich kombinácia:
  - riadený zápis dát na konkrétne miesto v pamäti,
  - prinútenie operačného systému nesprávne spracovať dátové typy.

Pretečenie zásobníka (pamäte) (pokr.)

- Pri jazykoch a prostrediach s lepšou kontrolou nad dátovými typmi („strongly-typed programming languages and environments“), charakter týchto techník znemožňuje výskyt útokov na pretečenia zásobníka.

Techniky prevencie

- Niekoľko všeobecných techník predchádzania pretečeniu zásobníka zahŕňa:
  - auditovanie kódu (automatické, alebo manuálne)
  - školenie vývojárov – správa miesta alokovaného pre premenné v pamäti (bounds checking), používanie potenciálne nebezpečných funkcií, skupinové štandardy,
  - nespúšťateľné zásobníky (Non-executable stacks) – veľa operačných systémov má podporu tejto funkcionality,
  - kompilačné nástroje – napríklad nástroje StackShield, StackGuard a Libsafe,

Techniky prevencie (pokr.)

- bezpečné funkcie – používanie strncat namiesto strcat, strncopy namiesto strcopy atď.,
- aktualizácie – pravidelná aktualizácia webových a aplikačných serverov a prehľad o aktuálnych publikovaných chybových reportoch týkajúcich sa aplikácií na ktorých je závislý kód,
- Pravidelné skenovanie aplikácií s niektorým s dostupných skenerov, ktoré sú schopné detegovať chyby pretečenia zásobníka v serverových produktoch a proprietárnych webových aplikáciách.

Atomickosť operácií a race conditions

- V databázových systémoch je nedeliteľnosť (alebo aj atomickosť) jedna z vlastností ACID transakcie. ACID transakcia, je transakcia ktorá spĺňa nasledovné požiadavky:
  - Atomickosť
    - *Vyžaduje, aby každá transakcia bola spracovaná spôsobom „všetko alebo nič“. Ak nezbehne časť transakcie, nezbehne ani transakcia ako celok a databáza zostane nezmenená.*
  - Konzistentnosť
    - *Táto vlastnosť zabezpečuje to, že každá transakcia presúva databázu z jedného validného stavu do iného validného stavu.*

Atomickosť operácií a race conditions (pokr.)

- Izolácia
  - *Táto vlastnosť zabezpečuje to, že súčasne prebiehajúce transakcie tranzakcie presunú systém do stavu, ktorý by nastal ak by boli tieto transakcie vykonané sériovo, t.j. jedna po druhej.*
- Trvalosť
  - *Trvalosť znamená, že keď už raz bola transakcia potvrdená a vykonaná (commit), zostane v platnosti aj v prípade výpadku prúdu, havárie alebo prípadných chýb.*

Atomickosť operácií a race conditions (pokr.)

- V atomickej transakcii, je rad operácií nad databázou vykonaný tak, že buď sú vykonané všetky, alebo nedôjde k žiadnej.
- Záruka atomicity bráni tomu aby došlo k aktualizácia databázy len čiastočne, čo môže spôsobiť väčšie problémy, než odmietnuť celý rad operácií úplne.
- http://en.wikipedia.org/wiki/ACID

Príklad č.1

- Príkladom atomicity môže byť objednávka letenky, kde sú vyžadované dve akcie: rezervovať si miesto a platba. Potenciálny cestujúci musí buď:
  - 1. aj si rezervovať miesto aj zaplatiť, alebo
  - 2. ani si nerezervovať miesto ani nezaplatiť.
- Rezervačný systém nepovažuje za prijateľné pre zákazníka zaplatiť za letenku bez zaručenia sedadla, ani rezerváciu sedadla bez úspešnej platby.

Príklad č.2

- Povedzme, že chce niekto previesť určité množstvo peňazí z jedného účtu na druhý. Iniciuje prevod, peniaze sa z jedného účtu odčítajú, ale ak predtým ako sa pripočítajú k druhému účtu nastane chyba a program sa preruší, klient by mohol prísť o túto sumu. V správne ošetrenej aplikácii, ak dôjde k takémuto zlyhaniu, potom kvôli atomicite, bude čiastka prevedená buď úplne, alebo sa prevod ani nespustí. Týmto spôsobom nedeliteľnosť chráni používateľa aby kvôli chybe v transakcii nedošlo ku strate peňazí.

Race condition

- Race condition je druh chyby v softvéri, ku ktorej môže dôjsť v dôsledku toho, že aplikácia predpokladá, že určité operácie prebehnú v určitom poradí. Ak to však nie je pravda, a zmení sa poradie týchto operácií, aplikácia môže zostať v nekonzistentnom stave, čo môže mať za následok porušenie integrity údajov alebo aj prerušenie behu aplikácie.
- Race condition situácia môže mať aj bezpečnostné implikácie a môže poslúžiť potenciálnemu útočníkovi napríklad na manipulácie aplikácie alebo obídenie rôznych bezpečnostných opatrení.
- Chybou je, že aplikácia niečo predpokladá, ale neoveruje si, či je predpoklad splnený, race condition túto chybu využíva.

Zneužitie prístupových práv

- Často majú aplikácie oveľa viac prístupových práv a privilégií než v skutočnosti potrebujú na svoju činnosť.
- Aplikácia existuje preto, aby plnila nejaké úlohy v konečnom dôsledku pre používateľa, aplikácia nepotrebuje nejaké prístupové práva a privilégiá, tie potrebujú používatelia, aby mohli aplikáciu používať.
- Typickým prípadom je situácia, keď aplikácia beží s administrátorskými oprávneniami.

Zneužitie prístupových práv (pokr.)

- Z tohto dôvodu potom dochádza k zneužitiu prístupových práv a to hlavne v dvoch situáciách:
  - Po úspešnom zneužití chyby v aplikácii má útočník širšie možnosti ďalšej eskalácie kompromitácie údajov, samotnej aplikácie alebo systému na ktorom aplikácia beží
  - Samotná aplikácia môže poskytovať možnosti čítania a manipulácie údajov, ku ktorým by používateľ (alebo útočník) za bežných okolností nemal prístup.

Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Opatrenia na zníženie dopadov aplikačných chýb

- V ideálnom prípade by softvér nemal obsahovať žiadne bezpečnostné nedostatky. Žijeme však v reálnom svete, kde je väčšinou proces návrhu a vývoja rôznym spôsobom limitovaný.
- Preto je rozumné predpokladať, že každá aplikácia v konečnom dôsledku obsahuje zraniteľnosti, ktoré zatiaľ neboli identifikované.

KPMG
cutting through complexity™

Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Opatrenia na zníženie dopadov aplikačných chýb

- Existuje riziko, že po nasadení našej aplikácie do prevádzky, v nej môže neskôr niekto (potenciálny útočník) objaviť zraniteľnosť, ktorú aj následne úspešne zneužije.
- Na takýto prípad sa vieme pripraviť využitím rôznych techník a nástrojov, ktoré zmiernia dopad takéhoto útoku.

KPMG
cutting through complexity™

Opatrenia na zníženie dopadov aplikačných chýb

- Medzi uvedené techniky a nástroje patria:
  - Separácia oprávnení (Privilege separation)
  - Modularizácia (Kompartmentalizácia)
  - Obmedzenie prístupu k aplikáciám
  - Obmedzenie prístupových práv aplikácie
  - Uväznenie aplikácie (Jailing)
  - Aplikačné firewally

Separácia oprávnení (Privilege separation)

- Separácia oprávnení je technika používaná pri programovaní aplikácií a v počítačovej bezpečnosti, pri ktorej je program rozdelený do častí (modulov), ktoré majú pridelené len tie oprávnenia, ktoré potrebujú na plnenie určitej úlohy.
- Používa sa pre zmiernenie potenciálnych škôd pri úspešnom útoku na aplikáciu.

Separácia oprávnení (Privilege separation) – pokr.

- Bežnou metódou pre realizáciu separácie oprávnení, je rozdelenie (pomocou systémovej funkcie „fork“) bežiaceho procesu na dve oddelené procesy.
- Hlavný program (väčší) sa vzdá vysokých oprávnení a menší program si tieto vysoké oprávnenia uchová, aby mohol neskôr vykonať určité privilegované operácie.

Separácia oprávnení (Privilege separation) – pokr.

- Menší program (ten s vysokými oprávneniami) je vytvorený tak, aby čistým spôsobom (t.j. jednoducho a prehľadne) vykonával privilegované operácie a má menej rozhraní a jednoduchšie rozhrania, než druhý väčší program vykonávajúci všetky ostatné aktivity.
- Takýto návrh zvyšuje pravdepodobnosť odhalenia prípadných zraniteľností v privilegovanej časti kódu.

Separácia oprávnení (Privilege separation) – pokr.

- To má za následok, že úspešná kompromitácia privilegovaného kódu, je oveľa menej pravdepodobná než v prípade zvyšku kódu.
- Obe časti pôvodného procesu potom komunikujú napr. cez dvojicu socketov.
- A napriek tomu, že bude táto dvojica programov schopná vykonávať aj privilegované operácie, následný úspešný útok na väčší program potom poskytne útočníkovi len minimálny prístup.

Separácia oprávnení (Privilege separation) (pokr.)

- Separácia oprávnení sa pod UNIX-ovými operačnými systémami tradične vykonáva rozlišovaním medzi skutočnými ID používateľa / ID skupiny a efektívnymi ID používateľa / ID skupiny pomocou setuid (2) / setgid (2) a súvisiacich systémových volaní, ktoré boli špecifikované štandardom POSIX.
- Ak sú však v kóde aplikácie nesprávne umiestnené, môžu byť takto vytvorené bezpečnostné medzery zneužité na rozsiahlu kompromitáciu aplikácie prípadne systému.

Separácia oprávnení (Privilege separation) (pokr.)

- Mnoho démonov sieťových služieb musí vykonať určité privilegované operácie, napr. otvoriť tzv. RAW sockety (sieťové rozhranie pre nízko-úrovňovú komunikáciu) alebo sieťový socket pre internetovú komunikáciu na nižších vyhradených portoch.
- Taktiež rôzne administratívne nástroje môžu počas svojho behu vyžadovať špeciálne oprávnenia.

Separácia oprávnení (Privilege separation) (pokr.)

- Takýto softvér zvykne separáciu oprávnení riešiť takým spôsobom, že sa vykonaní kritických operácií sám vzdá privilegovaných oprávnení a to tým spôsobom, že zmení používateľa pod ktorým beží na nejaký neprivilegovaný účet.
- Táto akcia je pod Unix-ovými operačnými systémami známa ako vzdanie sa administratívnych práv (dropping root).
- Neprivilegovaná časť zvyčajne beží pod používateľom "nobody" alebo obdobným samostatným používateľským účtom.

Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Otázky?

KPMG
cutting through complexity™