

Bezpečnosť webových aplikácií

RNDr. Richard Ostertág, PhD. (ostertag@dcs.fmph.uniba.sk)

18. marca 2021

Katedra informatiky
Univerzita Komenského, Bratislava

Éra webových sídiel (web sites – Web 1.0)

- statické dokumenty
- prehliadač slúžil iba na stiahnutie a zobrazenie týchto dokumentov
- jednosmerný tok „zaujímavej“ informácie (server $\rightarrow$ prehliadač)
- väčšinou bez autentifikácie používateľov
- každý používateľ: rovnaké zaobchádzanie, rovnaký obsah
- používatelia nevytvárajú prezentovaný obsah
- bezpečnostné hrozby využívali hlavne zraniteľnosti vo web-serveri
- kompromitovaný server
  - neznamenal únik citlivých informácií (všetky boli aj tak na webe)
  - modifikácia obsahu
  - využitie úložného priestoru a konektivity servera na šírenie warez-u

Éra webových aplikácií (web applications – Web 2.0)

- interaktívne dynamické aplikácie
- prehliadač sa stáva operačným systémom v ktorom beží webová aplikácia
- obojsmerný tok „zaujímavej“ informácie (server $\leftrightarrow$ prehliadač)
- obvykle existuje autentifikácia používateľov (registrácia, prihlásenie)
- každý používateľ: rôzne zaobchádzanie, iný prispôsobený obsah
- používatelia vytvárajú prezentovaný obsah
- bezpečnostné hrozby využívajú aj zraniteľnosti v samotnej webovej aplikácii
- kompromitovaný server / web aplikácia
  - únik citlivých informácií (osobné údaje, čísla kreditných kariet)
  - modifikácia obsahu (defraudácia peňazí, útoky na iných používateľov)
  - využitie prenosovej šírky pásma, výpočtového výkonu alebo úložného priestoru servera (vytváranie botnetov pre rozosielanie spamu alebo DDoS útoky)

Funkcie internetových web aplikácií

- Internetové obchodné domy
  - Amazon, Alza, Hej
- Sociálne siete
  - Facebook, Twitter, Second life
- Internet banking
  - Slovenská sporiteľna, Tatrabanka, VÚB
- Internetové vyhľadávače
  - Google, Bing, Baidu
- Internetové aukcie
  - eBay, Aukro
- Internetové média, TV a rádiá
  - YouTube, Netflix, Hulu, Spotify, Deezer
- Internetové kasína
  - bwin
- Internetové kasína
  - bwin, eTipos, PokerStars
- Web logs
  - Blogger, Write.as
- Web mail
  - Gmail, Outlook.com, GMX
- Internetové médiá
  - CNN, SME
- Internetové encyklopédie
  - Wikipédia, OEIS.org

Funkcie intranetových web aplikácií

- Aplikácia pre oddelenie ľudských zdrojov
  - platové podmienky
  - nábor nových zamestnancov
  - disciplinárne opatrenia
- Administratívne rozhrania kľúčovej infraštruktúry
  - web server
  - mail server
  - server pre virtuálne stroje
- Softvér pre spoluprácu v tíme
  - zdieľanie dokumentov
  - riadenie projektov
- Obchodné aplikácie
  - Podnikové plánovanie zdrojov (ERP)
  - Správa vzťahov so zákazníkmi (CRM)
- Softvérové služby
  - e-mailový klient
- Tradičné desktopové aplikácie
  - textový editor
  - tabuľkový procesor
- Vývojové nástroje
  - správa verzií zdrojového kódu
  - automatizované testovanie
  - code review

Desať najčastejších zraniteľností webových aplikácií[1]

1. Vloženie kódu – code injection
   - vloženie nedôveryhodných dát do príkazu pre interpreter, čím sa zmení jeho význam
2. Chybná autentifikácia – broken authentication
   - zlá implementácia autentifikácie používateľa alebo správy sedenia (session) umožňujúca útočníkovi dočasne alebo trvalo získať identitu iného používateľa
3. Vystavenie citlivých dát – sensitive data exposure
   - nedostatočná ochrana dát (napr. šifrovaním) pri prenose medzi prehliadačom a serverom (alebo pri ich uložení) umožňuje útočníkovi tieto dáta ukradnúť/upraviť
4. XML External Entities (XXE)
   - staršie a/alebo zle nastavené XML procesory môžu vyhodnotiť externé odkazy v definíciách entít a tak sprístupniť lokálne súbory alebo dokonca spustiť kód

[1] Zdroj: OWASP Top Ten z roku 2017.

Desať najčastejších zraniteľností webových aplikácií

5. Chybná kontrola prístupu – broken access control
   - obmedzenia prístupu pre používateľov sa často nevynucujú správne
6. Chybná konfigurácia zabezpečenia
   - obvykle dôsledok nevhodných počiatočných nastavení, podrobných chybových hlásení obsahujúcich citlivé informácie, ...
7. Cross-Site Scripting – XSS
   - aplikácia vkladá nedôveryhodné dáta bez riadnej kontroly do web stránky
8. Nedostatočne zabezpečená deserializácia
   - často vedie k vzdialenému spusteniu kódu (Java, .NET, JavaScript, ...)
9. Používanie komponentov so známymi zraniteľnosťami
   - komponenty sa spúšťajú s rovnakými oprávneniami ako aplikácia $\rightarrow$ zneužitie zraniteľného komponentu môže uľahčiť stratu údajov alebo prevzatie servera
10. Nedostatočné záznamy (logging) a monitorovanie
    - čas na zistenie prieniku $>$ 200 dní, (zvyčajne ho zistia až externé strany)

SSL, PCI – falošný pocit bezpečia

- používatelia sú si vedomí bezpečnostných nedostatkov web aplikácií
- veľa web aplikácií tvrdí, že sú bezpečné, lebo používajú SSL[2]:

  This site is absolutely secure. It has been designed to use 128-bit Secure Socket Layer (SSL) technology to prevent unauthorized users from viewing any of your information. You may use this site with peace of mind that your data is safe with us.
- tiež zvyknú uvádzať, že vyhovujú Payment Card Industry (PCI) štandardom:
  We take security very seriously. Our web site is scanned daily to ensure that we remain PCI compliant and safe from hackers.
  - používajte a udržiavajte firewall
  - nepoužívajte výrobcom predvolené heslá
  - šifrujte dáta prenášané cez verejné siete
  - udržiavať bezpečnostnú politiku, …

[2] Zdroj: Dafydd Stuttard, Marcus Pinto: The Web Application Hacker's Handbook

Základný bezpečnostný problém

- útočník môže odoslať akúkoľvek odpoveď
  - bezpečnostné kontroly vykonávané na strane klienta je možné obísť
  - pre prístup k aplikácii sa nemusí používať webový prehliadač
  - útočník (na strane klienta) môže:
    - čítať, pozmeňovať, odstraňovať alebo opakovane používať všetky dáta, ktoré dostane od serveru alebo web-prehliadač posiela na server
    - generovať nové dáta a vkladať ich do komunikácie
    - potlačiť, modifikovať, opakovať akékoľvek operácie vykonávané na strane klienta
  - útočník môže manipulovať so všetkými dátami, s ktorými klient pracuje:
    - URL (časti cesty, GET parametre)
    - polia formulárov (aj skryté)
    - zakomentované časti HTML dokumentov
    - skripty (prezerať, meniť)
    - cookies a iné údaje v hlavičkách HTTP (napr. session ID, User-Agent)
- je nutné predpokladať, že všetky vstupy sú potenciálne nebezpečné

Kľúčové faktory základného bezpečnostného problému

- nedostatočné bezpečnostné povedomie
  - problematika zabezpečenia webových aplikácií je mladá (vs. siete, OS)
- vlastný vývoj
  - veľa webových aplikácií je vyvinutých zamestnancami rôznej kvalifikácie
  - každá aplikácia je iná a môže obsahovať svoje vlastné jedinečné vady
- zavádzajúca jednoduchosť
  - začínajúci programátor môže vytvoriť funkčnú webovú aplikáciu od nuly
  - produkovať funkčný kód a produkovať bezpečný kód je ale rozdiel
- nové hrozby pre web aplikácie pribúdajú rýchlejše ako pre staršie technológie
- obmedzené zdroje (čas, peniaze, ľudia, …)
  - funkčnosť aplikácie má prednosť pred jej bezpečnosťou
- „zneužitie“ technológií
  - mnohé zo základných technológií vo webových aplikáciách sú využívané ďaleko za pôvodne účely, čo vedie k ich bezpečnostnej zraniteľnosti

Základné obranné mechanizmy

- kontrola prístupu používateľov
  - zabrániť používateľom v získaní neoprávneného prístupu
- kontrola používateľských vstupov
  - zabrániť nežiaducemu správaniu aplikácie pri „zákernom“ vstupe
- zvládnutie útoku
  - korektná funkčnosť aj v prípade priameho útoku
  - obranné a útočné opatrenia pre odrazenie útoku
- monitorovanie aplikácie
  - správca môže okamžite zasiahnuť

Kontrola prístupu používateľov

- základné komponenty:
  - identifikácia a autentifikácia
  - správa relácií (session management)
    web aplikácia vydá používateľovi token identifikujúci jeho reláciu
  - kontrola prístupu
- chyba v ktorejkoľvek komponente môže viesť k získaniu neoprávneného prístupu

Identifikácia a autentifikácia

- identifikácia
  - najčastejšie pomocou prihlasovacieho mena
- autentifikácia
  - najčastejšie pomocou hesla
- utočník môže
  - získať prihlasovacie mená
  - získať heslá
  - obísť prihlasovaciu funkciu
    - vďaka chybe v návrhu

Identifikácia a autentifikácia (klasické problémy)

- slabé heslá (krátke, malá abeceda, uhádnuteľné)
  - krátke
  - malá abeceda
  - slovníkové
  - uhádnuteľné (dátum narodenia)
    - kontrola kvality (napr. aj slovníkový útok)
- rovnaké heslá do rôznych systémov
- neobmieňanie hesiel
  - kontrola veku hesla
- keylogger
  - virtuálna klávesnica
  - jednorázové heslá
  - mechanizmy výzva-odpoveď (zero knowledge proofs)
- neskorý prechod z HTTP na HTTPS
- pri neúspešnom prihlásení nerozlišovať zlé meno od zlého hesla

Správa relácií – session hijacking[3]

- prezradenie identifikátoru relácie
  - odpočúvaním komunikácie
  - hlavička Referer pri prechode na iné stránky
  - história prehliadača
  - prístup k používateľovmu počítaču a prečítanie uložených cookies, …
- uhádnutie identifikátoru relácie
  - identifikátor je generovaný predikovateľným spôsobom (napríklad jednoduchá aritmetická postupnosť)
  - identifikátor má malý rozsah možných hodnôt
- možná ochrana (nie je jednoduchá a 100 % účinná):
  - identifikátor nie je len náhodne vygenerované číslo
  - má k sebe pripojený aj hash IP adresy servera a klienta, User-agent hlavičky klienta a nejakej tajnej hodnoty
  - útočník potom nemôže jednoducho použiť ukradnutý identifikátor

[3] útočník zistí ID inej relácie a cezeň do nej vstúpi a pracuje pod inou identitou

Správa relácií – cross-site request forgery (CSRF)

- cross-site request forgery, aka. a one-click attack alebo session riding
- session hijacking vyžaduje, aby útočník ukradol/uhádol session ID
- session riding nevyžaduje od útočníka znalosť session ID

- útočník presvedčí používateľa aby poslal ním skonštruovanú žiadosť
  - presvedčí ho aby klikol na nejakú ním zostavenú linku (napr. v diskusii)
  - využitie obrázkov na odoslanie žiadosti

- možná ochrana:
  - nepoužívanie cookies na ukladanie session ID
  - vkladať session ID priamo do URL
  - náhodne generovať a vkladať autorizačné tokeny do každej akčnej URL

Kontrola prístupu

- prihlásení používatelia môžu mať prístup iba k určitej časti webu
- útočník môže získať neoprávnený prístup využitím chybného predpokladu programátorov o tom, ako budú používatelia komunikovať s aplikáciou
- URL tampering – pozmeňovanie častí (najmä GET parametrov) existujúcich URL
  - ak v aplikácii útočník narazí na URL v tvare:
    https://www.app.sk/zaznam.php?id=1234
  - môže skúsiť zadať URL v tvare:
    https://www.app.sk/zaznam.php?id=1235
- Forceful browsing – vytváranie nových URL a vynútený prechod na ne
  - útočník môže skúsiť zadať URL v tvare:
    https://www.app.sk/zaznam.php.old
  - prípadne môže skúsiť zadať:
    https://www.app.sk/admin.php
  - server sa spolieha na to, že keď takú URL neposlal, tak ani „nemôže“ byť taká URL zo strany klienta požadovaná

Kontrola vstupov – rôzne prístupy

- odmietni nebezpečné vstupy

  vyhľadáva známe reťazce používané pri útokoch
- akceptuj bezpečné (povoluje len určite neškodné vstupy)
- ošetrenie vstupu
  - odstráň potenciálne nebezpečné postupnosti znakov (napr. <script>)
  - ’<scr<script>ipt>’ ?
  - ’+ADw-script+AD4-’ ? (UTF-7, dnes už podporované iba IE 11)
- bezpečná práca so vstupom
  - parametrizované dotazy do databázy
- sémantické kontroly
  - vstupné dáta sú syntakticky korektné (napríklad číslo účtu)
  - sémanticky však nie (cudzie číslo účtu)
- vstup musí byť kontrolovaný na strane servera
  - aj keď sa pre lepší UX kontroluje aj na strane klienta
  - najlepšie opakovane kontrolovať v každej časti aplikácie

XPath injection

- //user[name/text()='Janko Hrasko' and password/text()='passwd']/account/text()
- vstup: ' or 1=1 or ''='

  //user[name/text()='' or 1=1 or ''='' and password/text()='passwd']/account/text()
- vstup: NoSuchUser'] | P | //user[name/text()='NoSuchUser

  //user[name/text()='NoSuchUser'] | P |
  //user[name/text()='NoSuchUser' and password/text()='passwd']/account/text()

  prvá a posledná časť nič nevrátia $\Longrightarrow$ výsledkom je vyhodnotenie P
- pri prístupe k XML cez XPath nie je možné aplikovať prístupové práva (v rámci tagov)
  - preto druhým z uvedených spôsobov sa dá získať celý XML dokument
  - naproti tomu pri SQL injection bude útočník obmedzený iba na tabuľky, kam má daná aplikácia prístup

Cookie relácie

- nepoužitie HttpOnly – zabránilo by skriptom na strane klienta v prístupe ku cookie
- príliš široká doména/cesta cookie relácie
  - útok prostredníctvom iných aplikácií zdieľajúcich rovnakú doménu/cestu
- trvalá cookie relácie
  - zostáva v platnosti aj po zatvorení prehliadača používateľom
  - často sa používa ako súčasť funkcie „Pamätaj si ma“
- neposlanie cez SSL
  - Moderné webové prehliadače podporujú secure príznak pre každú cookie.
    - Ak je tento príznak nastavený, prehliadač odošle cookie iba cez protokol HTTPS.
  - Ak aplikácia používa HTTPS aj HTTP bez secure príznaku,
    - potom sa v nasledujúcich HTTP požiadavkách odošlú aj cookie vytvorené cez HTTPS.
- zakázané
  - Ak program nepoužíva cookie pre uloženie ID relácie,
    - potom sa ID odovzdáva ako parameter požiadavky alebo ako súčasť URL.
  - Ponecháva otvorené dvere pre fixáciu relácie a session hijacking.

Chybná konfigurácia systému

- Porušenie pravidla minimálnych privilégií (least privilege)
- Správa hesiel
  - Prázdne, pevne nastavené (hard-coded) alebo štandardné heslá
  - Heslo v konfiguračnom súbore
    - Nechránené
    - Chránené slabou kryptografiou
- Únik informácií o systéme
- Chýba vlastná chybová stránka
- Nesprávne použitie SSL
  - Slabé šifry alebo protokoly
  - Samopodpísané certifikáty

Nedostatočné bezpečné programovanie

- Pretečenie vyrovnávacej pamäte, celých čísiel, formátovací reťazec
- Práca s výnimkami
  - Prázdny catch blok, t.j. ignorovanie výnimiek
  - Príliš široký catch blok (zložitosť kód na spracovanie chýb)
- Nedostatočná náhodnosť
- Nezačatie novej relácie po autentifikácii
- Časová závislosť pri prístupe k súboru (TOCTOU)
- Smerníky
  - Viacnásobné uvoľnenie
  - Použitie po uvoľnení
  - Únik pamäte (memory leak)
  - Dereferencovanie null
- Zabudnutý testovací/ladiaci kód

Zvládnutie útoku

- vývojári musia predpokladať, že aplikácia bude cieľom útoku
- zvládnutie útoku vyžaduje
  - ošetrenie chýb
  - vytváranie záznamov (audit logs)
  - varovanie správcov
  - reakciu na útok

Zvládnutie útoku – ošetrenie chýb

- v aplikácii sa nevyhnutne objavia neočakávané chyby
  - je veľmi ťažké identifikovať každý možný spôsob interakcie útočníka s aplikáciou
- aplikácia by mala zvládnuť neočakávané chyby
  - zotavením z chyby
  - prezentovaním vhodnej chybovej správy
    - správca – podrobná
    - používateľ – stručná

Zvládnutie útoku – vytváranie záznamov

- neoceniteľný zdroj informácie pri vyšetrovaní pokusov o prienik
- pomáhajú zistiť:
  - čo sa stalo
  - aké zraniteľnosti boli využité
  - k akým dátam sa útočník dostal
  - útočníkovu identitu
- nasledovné udalosti by sa mali vždy zaznamenávať:
  - (ne)úspešný pokus o prihlásenie
  - zmena hesla
  - dôležité transakcie (napr. presun finančných prostriedkov)
  - zamietnutie prístupu
  - odpovede/požiadavky klienta pokúšajúce sa o známe útoky

Zvládnutie útoku – varovanie správcov

- správcovia môžu online riešiť problém (namiesto spätnej offline analýzy)
  - blokovanie útočníkovej IP adresy
  - nastraženie pasce
- varovanie by malo prísť pri
  - netypickom použití
    - veľký počet požiadaviek z jednej IP v krátkom čase
    - veľký počet presunov finančných prostriedkov z rôznych účtov na jeden účet
  - odpovediach/požiadavkach klienta
    - pokúšajúce sa o známe útoky
    - so zmenenými údajmi skrytými pred bežným používateľom

Zvládnutie útoku – reakcia na útok

- systém môže reagovať automaticky
  - ukončením útočníkovej relácie
  - spomalením odpovedí na útočníkove IP adresy
  - blokovaním útočníkových IP adries
  - varovaním napadnutého používateľa
- získa sa tým čas pre správcov