Zásady bezpečného programovania v PHP
Zabezpečenie LAMP konfigurácie

Ing. Pavol Lupták, CISSP, CEH
nethemba@nethemba.com

Kontrola vstupu

- nutnosť kontrolovať všetky vstupné parametre (nielen bezprostredne zadané užívateľom, aj User-Agent, http referer apod), použiť **whitelist**
- ochrana voči SQL injection **addslashes(), stripslashes(), mysql_real_escape_string(),** XSS útokom **htmlspecialchars()**
- nepoužívať **strip_tags()** (neodstraňuje atribúty, jedine ak chceme čisto plaintextový výstup)

PHPIDS

- použiť PHPIDS na všetky užívateľské vstupy
- jednoduchá integrácia do aplikácie
- „váhovanie“ užívateľského vstupu (numerical impact rating) – možnosť sa rozhodnúť, čo so vstupom ďalej spraviť – len zalogovať, zablokovať žiadosť, notifikovať admina..

Globálne premenné

- PHP nedeklaruje premenné, len definuje – táto vlastnosť môže byť zneužitá útočníkom
- nepoužívať globálne premenné (register_globals = off)
- vstupné dáta používať z $_GET, $_POST, $_REQUEST, $_COOKIES ...

magic_quotes_gpc

- „escapuje“ uvodzovky v „queries“
- veľmi málo účinný bezpečnostný workaround:
- programátori sa implicitne spoliehajú na ošetrenie vstupov PHP
- dá sa ľahko obísť ak útočník použije napríklad hexadecimálnu reprezentáciu a úvodzovky úplne vynechá (0x61646D696E = 'root')

PHP Safe Mode

- vypína použitie nebezpečných funkcií (system(), exec(), popen())
- definuje adresár spustiteľných súborov (safe_mode_exec_dir)
- definuje strom povolených súborov (open_base_dir)
- vypína nebezpečné funkcie/triedy (disable_functions, disable_classes)

Prepared SQL statements

- SQL queries vznikajú „lepením“, vstupy nie sú veľakrát poriadne ošetrené
- je potrebné použiť PHP5 mysqli „prepared statements“ **mysqli_stmt_prepare()** / **mysqli_stmt_bind_param()** / **mysqli_stmt_bind_result()**
- defacto odpadne hrozba SQL injection útokov

MySQL5 stored procedúry

- Použiť MySQL5 Stored procedúry
- možnosť „n-tier“ aplikácie – lepšia separácia dátovej vrstvy od aplikačnej
- útočník pri kompromitovaní aplikácie nezíska fyzický prístup k tabuľkám, ale len k definovaným stored procedúram

Klasické bezpečnostné problémy I.

- validácia formulárov len cez JavaScript (nie na serveri!)
- prenos/modifikácia dát v HIDDEN fieldoch formulárov (nikdy neveriť užívateľskému vstupu!)
- staré verzie PHP zdrojových kódov, DB dumpov, patchov na produkčných serveroch

Klasické bezpečnostné problémy II.

- všetky citlivé dáta by mali byť mimo webrootu
- zapnutá http basic autentizácia (možnosť rýchleho útoku hrubou silou – použiť digest, resp. klientské certifikáty)

Logovanie

- nevyhnutné, aby bola dodržaná „accountability“
- logovať krátko a prehľadne
- dátum, čas, IP adresa, miesto v kódu, typ hlášky
- logy offline uchovávať, príp. na write-once médium, digitálne podpisovať
- pozor na log injection !

Ako zabezpečiť LAMP systém I.

- Apache chroot (buď libapache2-mod-chroot, resp. SecChrootDir z balíčka mod_security)
- Separácia pomocou virtualizačných prostriedkov (XEN, VMWare)
- **Úpravy zo strany aplikácie:** prístup k MySQL musí byť cez TCP, nie Unixové sockety, pošta musí byť posielaná cez SMTP (napr. Pear Mail) nie cez mail(), ktorý volá /usr/sbin/sendmail

Ako zabezpečiť LAMP systém II.

- Použiť WAF (Web Application Firewall) – veľmi robustný opensource WAF je mod_security
- dokáže zablokovať 70-90% všetkých útokov na vstupy aplikácie (XSS, SQL/LDAP injection, ..)
- **Úpravy zo strany aplikácie:** žiadne, ak aplikácia nepoužíva na vstupoch špeciálne metaznaky, veľké, ak používa zvrhlé vstupy:-)

Ako zabezpečiť LAMP systém III.

- zapnúť PHP Safe mode, znemožniť vykonávanie nebezpečných funkcií (system(), exec(), popen(), phpinfo(), ..)
- vypnúť URL fopen, register_globals, zapnúť quotovania queries (magic_quotes_gpc), definovať open_basedir, safe_mod_exec_dir, safe_mod_include_dir
- **Úpravy zo strany aplikácie:** relatívne veľké

Ako zabezpečiť LAMP systém IV.

- Použiť PHP Suhosin / Core GRASP
- Engine / Runtime / Session ochrana
- Filtrovacie / logovacie možnosti
- účinna ochrana voči SQL injection útokom, experimentálna heuristická detekcia
- **Úpravy zo strany aplikácie:** malé (nutné otestovať)

Ako zabezpečiť LAMP systém V.

- použiť suPHP
- umožňuje púšťať PHP skripty s oprávneniami ich vlastníka, možnosť chrootu
- lepšia segregácia virtualhostov – útočník, keď kompromituje jeden web nezíska automaticky prístup k ďalšiemu
- **Úpravy zo strany aplikácie:** minimálne

Ako zabezpečiť LAMP systém VI.

- nasadiť NSA SELinux (Secure-Enhanced)
- implementácia MAC použitím LSM nad Linuxovým jadrom využívajúca princíp „least privilege“
- MAC = DTE + RBAC + MLS
- **Úpravy zo strany aplikácie:** relatívne malé, ladenie politiky je ale náročné, nutnosť detailného testovania

Ideálny stav bezpečnosti aplikácie

- ošetrovania všetkých vstupov, 3rd layer architektúra aplikácie (stored procedúry) + prepared statements + PHPIDS + Apache chroot + mod_security WAF + PHP Safe mode + PHP Safe settings + PHP Suhosin + Core GRASP + suPHP + detailné logovanie + NSA SELinux so zapnutým RBAC/DTE/MLS

Vďaka za pozornosť