THE ITALIAN RESEARCH & EDUCATION NETWORK | Consortium GARR

Tutorial Sicurezza

Vulnerabilità nelle Web Applications

Andrea Lora >> Tutorial Sicurezza #garrws16

GARR WORKSHOP 2016 | Roma - 18-21/04/2016

Perché attaccano?

- Le motivazioni che portano ad un attacco possono essere le più disparate che potremmo riassumere con il classico
- «For Fun and Profit»
  - Da una parte ci sono «script kiddies» che sfruttano i google dorks per eseguire defacement di siti, a volte inserendo un messaggio di propaganda
    - Sono i più facili da rilevare a causa della perdita di funzionalità del sito: il sistema di monitoring comincia a segnalare una serie di errori 404 su pagine che 5 minuti prima c'erano o arriva una telefonata al gruppo IT : (
  - Dall'altra hacker che sanno quanto può essere performante la nostra rete, facendone un candidato perfetto per nodi di botnet
    - Più insidiosi da rilevare giacché non vengono interrotte le funzionalità del sito: il sistema di monitoring può segnalare anomalie nel sistema (alto uso CPU / Rete) oppure può arrivare una mail dal CERT del GARR
  - Ultima casistica è quella in cui l'attaccante cerca attivamente dati riservati contenuti sui server (documenti/mail)

Perché dovrebbero attaccare me?

- Nella stragrande maggioranza dei casi gli attacchi *non sono mirati,* ma derivano dai cosiddetti google dorks: ricerche fatte tramite google che mirano a trovare web applications che hanno bachi di sicurezza
  - Joomla e Wordpress (e relativi plugin) sono i bersagli privilegiati perché comunemente indicizzati
  - Quando non è necessario che un sito venga indicizzato il file robots.txt può aiutare, ma non tutti i motori di ricerca onorano le sue direttive
  - Esempio di google dork che cerca tutti i siti wordpress con un particolare plugin (Imagezoom):
    - filetype:php inurl:"/wp-content/plugins/wp-imagezoom" & inurl:"?id="

Database di vulnerabilità

- Numerosi siti mantengono dei database di vulnerabilità, con comodo motore di ricerca, sia in forma di advisory, sia con il codice per eseguire l'exploit della vulnerabilità.
- A volte contengono il google dork per cercare siti vulnerabili o i plugins metasploit per automatizzare l'attacco
  - https://packetstormsecurity.com/
    - Ha compiuto 15 anni di attività la settimana scorsa!
  - https://www.exploit-db.com/
    - Dai creatori di Backtrack/Kali

Vulnerabilità delle WebAPP

- Le vulnerabilità delle applicazioni web sono indicate nell'advisory e ricadono in determinate categorie:
  - LFI – Local File Inclusion
  - RFI – Remote File Inclusion
  - AFD – Arbitrary File Download
  - AB – Autentication Bypass
  - SQLI – SQL Injections
  - XSS – Cross Site Scripting
  - CSRF – Cross Site Request Forgery

Anatomia di una SQL Injection 1

- Supponiamo un form html per il login con due parametri: UserName e UserPass, e il seguente codice php
- uName = getRequestString("UserName");
  uPass = getRequestString("UserPass");
  sql = "SELECT * FROM Users WHERE Name ='" + uName + "' AND Pass ='" + uPass + "'";
- Se sql dopo la query contiene qualcosa (user e password corrette) allora login autorizzato

Anatomia di una SQL Injection 2

- Supponiamo che i parametri fossero
  - uName = andrea
  - uPass = pass
- La query sarebbe stata
  - SELECT * FROM Users WHERE Name ='andrea ' AND Pass ='pass'
- La query viene fatta al DB, se ritorna qualcosa allora i dati sono validati

Anatomia di una SQL Injection 3

- Visto che i parametri vengono presi della form abbiamo la possibilità di inserirne di arbitrari, per esempio:
  - uName = nonesiste' OR 1=1 --
  - uPass = pass
- La query a quel punto diventerebbe
  - SELECT * FROM Users WHERE Name ='nonesiste' OR 1=1 -- AND Pass ='pass'
- Considerando che i -- dicono a mysql di considerare ciò che segue come commento la query finale diventa una tautologia (1 è sempre = a 1), come se la query fosse
  - SELECT * FROM Users

Anatomia di una SQL Injection 4

- L'injection prima descritta è il classico esempio di Authentication Bypass
- Esistono altri tipi di Injection che in alcuni casi possono portare ad un dump completo del database
- Un esempio di queste injection sono le cosidette Time Based Blind Query
- Si concatena la query primaria con un comando SLEEP a sua volta concatenato con query che riguardano lo schema e i dati
- Un carattere alla volta (!) i dati vengono dumpati

SQL Injection - SQLMAP

- Sqlmap
  - Automatic SQL injection and database takeover tool
- Codice aperto su github // Molta documentazione
- Aiuta a scoprire (e a fare gli exploit) di sql injection
- In determinate condizioni può eseguire un dump completo del database (comporta centinaia di richieste HTTP/S)
- Flag principali
  - -u http://example.com/login.php (url bersaglio)
  - -forms (per l'analisi di forms)
  - -batch (batch mode, non interattivo)
  - -current-db (lavora sul DB dove c'è l'injection
  - -dump (mostra a schermo il contenuto del db) [!!!]

Enumeration

- Nmap è il re dei tool di discovery / enumeration, ma le funzioni di analisi del layer 7 non sono specializzate (di base)
- Sono stati scritti dei tool che interrogano i webserver alla ricerca di file e directory potenzialmente pericolosi
- Nikto è uno di questi, nella sua forma più semplice richiede solo l'host bersaglio
    - nikto.pl -host http://example.com

WPscan

- Un altro dei tool di ricerca di vulnerabilità, WPScan si occupa di analizzare un blog wordpress alla ricerca di bug presenti nell'engine, nei plugin e nelle templates
- Viene aggiornato costantemente con i nuovi bug scoperti
- Sintassi base:
    - ruby wpscan.rb --url www.example.com

PHP Web Shell

- In certe condizioni è possibile caricare sul server da attaccare dei file
- Le PHP Web shell consentono di operare comandi shell sul server remoto
- Questi comandi gireranno con gli stessi privilegi del demone che fa girare apache/php
  - mod_user / mod_itk / mod_chroot mitigano in qualche modo o del tutto l'efficacia di una php web shell a seconda del risultato che l'attaccante vuole ottenere
  - Una famosa web shell è http://phpfm.sourceforge.net/
- Alcune web shell sono create appositamente per evitare certe limitazioni del sistema (per esempio il fatto che il php.ini impedisca la funzione exec()) o per evadere i WAF

Privilege Escalation – Becoming Root

- A meno di grossi errori nella configurazione prendere il controllo di una web application non equivale a compromettere la macchina nella sua interezza
- Nel caso più frequente l'utente compromesso è quello che fa girare il demone http
- La scalata di privilegi avviene in contesti particolari, ed è causata principalmente da software di sistema non aggiornato che l'attaccante sfrutta per diventare root (Local privilege escalation) o da (appunto) problemi involontariamente introdotti dal sysadmin

Proteggere le web applications 1

- Alcune difese «classiche»
  - Aggiornamenti software in caso di prodotti terzi (wordpress / joomla e plugin relativi)
  - Utilizzo di librerie conosciute e ben sviluppate nel caso di prodotti sviluppati «in house» (framework php / librerie per le query mysql)

Proteggere le web applications 2

- Contro i day-0 ?
  - WAF – Web Application Firewall
  - L'implementazione opensource per apache/nginx è mod_security
  - Utilizza espressioni regolari e ruleset per individuare gli attacchi
  - Può essere configurato in maniera passiva (only detection/logging) o attiva (bloccante)
  - Da testare sulla propria applicazione web, può causare malfunzionamenti
  - Non è la panacea!

Proteggere le web applications 3

Opensource e Free

- OWASP ModSecurity Core Rule Set (CRS)
  - Community based
  - Disponibili su github
  - HTTP Protocol Protection
  - Real-time Blacklist Lookups
  - HTTP Denial of Service Protections
  - Generic Web Attack Protection
  - Error Detection and Hiding

Commerciali

- Commercial Rules from Trustwave SpiderLabs
  - Aggiornate quotidiamente
  - ~500$ / anno / istanza
  - IP Reputation
  - Web-based Malware Detection
  - Webshell/Backdoor Detection
  - Botnet Attack Detection
  - HTTP Denial

CTF – Capture The Flag

- Un tipo di «competizione» che riguarda la sicurezza informatica, ne esistono di vari tipi
  - Jeopardy
  - Attack-Defense
  - Mixed mode
- Nel nostro caso il più semplice, Jeopardy: nella directory /root esiste un file chiamato flag.txt, leggerne il contenuto equivale a terminare con successo il CTF

CTF – Capture The Flag

- Repository chiavi e cheatsheet
- http://bit.do/garrws16

CTF #garrws16 – Tips and tricks 1

- Eseguite una scansione di rete e un'enumerazione dei servizi (nmap/nikto)
- Trovato qualcosa di interessante? Cercate di bypassare il meccanismo di autenticazione
- Le autenticazioni HTTP sono molto difficili da forzare: a volte per andare da A a B si deve passare per C
- ../../../../ è un path valido in certe situazioni

CTF #garrws16 – Tips and tricks 2

- Un utente admin su wordpress non si crea necessariamente dalla sua interfaccia web
- Avere i privilegi di admin su wordpress permette di fare cose, che permettono a loro volta di fare più cose
- I file di log possono essere una miniera di informazioni

CTF #garrws16 – Tips and tricks 3

- "With great power comes great responsibility" vale anche per gli script
- L'espansione delle variabili di bash può presentare delle sorprese
- La pagina *man* di un comando non necessariamente è aggiornata
- A volte serve andare a leggere la documentazione direttamente sulla pagina web
- La sezione ***3.8*** pare essere particolarmente interessante