Global Information Assurance Certification Paper

PC Week Hack of 1999
Shawn Balestracci
February 26, 2001

In September of 1999, PC Week Magazine set up securelinux.hackpcweek.com and offered $1000 to the first person who could successfully hack into it. Many people tried, and four days later someone broke into the machine. A CGI exploit allowed a hacker going by the name of Jfs to get his foot in the door.

PC Week installed Red Hat Linux 6.0 with the latest version of Apache Web Server. They also implemented commonly used security products to fend off certain high level attacks. Their firewall blocked all ports except for port 80 (as it was to be a web server) and they ran a popular commercial banner ad system. The sample CGI scripts that come with apache were also disabled. They also specified that the server would provide Server Side Includes and mod_perl, however they failed to do this. This seemed like a reasonable approach, lock down the server with a firewall, run a minimal set of applications, and use a commercial, popular banner ad system. In 1999, this site had more thorough security than many other sites on the net, a scary thought.

How did it happen?

This hack began like most hacks, with the gathering of information. A port scan showed that the only thing open was the http server. The hacker scanned through the HTML of the web pages. He then queried the server and was able to gather that it was an Apache 1.3.6 server on Redhat. The server had the directories "/ /cgi-bin /photoads/ and /photoads/cgi-bin" The hacker then tried some well known CGI exploits, wwwboard, test-cgi, Count.cgi which were not available. Those CGI exploits came preinstalled with many versions of apache.

The hacker then turned to the /photoads/ directory With a web search, he found out that it is a commercial CGI package selling for $149. He found a friend that had a photoad installation, who then let Jfs take a loot at the source.

The hacker then simply looked at the default installation files, and attempted to retrieve the ads database which was at /photoads/ads_data.pl. This attempt was successful; it allowed the hacker to see the user passwords for their ads.

Next the hacker was able to gather some more information with an env.cgi script that was included with photoads (which is similar to test-cgi) and found the document root of /home/httpd/html.

The hacker then tried to exploit SSI and mod_perl. He looked for an unfiltered HTML variable, and found one in the post.cgi script, $ENV{'HTTP_REFERRER'}. However, the server wasn't running SSI or mod_perl.

Next the hacker went after the CGI scripts themselves looking for holes. Typical holes for perl scripts are in the open(), system(), and `` calls. The hacker found a few that used "open()" calls.

Most of the variables used in the open calls were defined in the configuration files. However $filename is extracted from the HTML variables on the form. There is some filtering on this variable, the script requires the filename to end in .gif or .jpg and also prevents "./../" (which would mean to go up a directory, we're trying to overwrite files here.) The hacker was able to get by this with help from an article in Phrack on perl CGI security.

The script also did some checks on the header for the GIF to be uploaded. This was easily circumvented by making sure that the 6th through the 9th byte of the file are 0. The script then renames the file that was uploaded to match the ad number. The key to get around this is to get the rename function to fail, as it has no error checking and would proceed as normal if it failed. How does the hacker cause rename to fail? He uses a ad number that is longer than 1024 bytes.

One last thing that the script requires is that the ad number already exist, so the hacker finds a way to create an ad with a number of his choosing, by again explioting another faulty input check and an embedded newline.

The hacker now has the ability to overwrite files with the permissions of the user "nobody." He tries to overwrite the main page but fails, because that file, index.html, is owned by a different user. The hacker then tries to overwrite another CGI file, advisory.cgi, and is successful. All the hacker needs to do now is upload a shell script that will let him execute his own arbitrary commands.

You can see below an example of Jfs's attempt at overwriting the main index file. Notice the long message number, which is designed to thwart the rename, the message (which is URL encoded), the use of the "\" to get around the scripts parameter checking, and the %00 NUL character at the end to save to an HTML file opposed to a GIF.

```
"http://securelinux.hackpcweek.com/photoads/cgi-
bin/photo.cgi?file=a.jpg&AdNum=11
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
```

```
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111111111111111
1111111111111111111111111111111111111111111111111111111111111111
11111111&DataFile=1&
Password=0&FILE_CONTENT=%47%49%46%38%39%61%c8%00%58%00%e6%0
0%00%0aJfs%20%20was%20
Here%20-
Details%20in%20the%20file%20called%20jfs(something)%20in%20
the%20photoads
%20directory%0a&FILE_NAME=/lala/\../../../../../../../home/
httpd/html/index.html
%00.gif"
```

The shell script idea fails, but the resourceful hacker uploads a stripped binary instead. The hacker now can do everything that the user "nobody" can do. Now that the hacker has the ability to create and run files, he searches for an exploit that would allow him to get root access. Finding one that gives him an interactive root shell, he modifies it to let him create a setuid root shell script. He places this in the /tmp directory. With this he's able to overwrite the main HTML file (as root is the superuser) and has won himself $1000.

Where did PC Week go wrong?

In setting up their server for this contest, their first mistake was not enabling the modules in the web server as they said they would in the contest specifications. Security wise, however, this was a smart move as they didn't have any need for server side includes, or mod_perl which can introduce their own vulnerabilities.

Since the hacker was able to get his foot into the door due to the photoad program, this is an obvious problem. PC Week failed to remove the sample script that came with Smart Photo Ads. One could also question their choice of a photoad system. Sure the source was available to them, but the source was never really scrutinized as people would for the most part purchase it, then simply set it up. Many people live with a false sense of security that the products that they spend significant amounts of money on are rigorous in security. A non commercial application might have been a better choice in this case, as the code is available to the public and is more likely to be under scrutiny by skilled developers. There's some debate as to whether or not open source products are more secure than closed source, it really boils down to the developers working on it.

PC Week also failed to evaluate the code of the Smart Photo Ad for vulnerabilities. They missed the lack of a filter on one of the user provided variables and didn't scrutinize the system calls: open() and rename(). There were also some file permission issues with these scripts as no file should be writable by user nobody, especially within the web tree.

PC Week did not install the latest security patches available at the time for Red Hat. These patches would have blocked the well known crontab exploit that the hacker used to get root access, as well as others. PC Week claimed that they wanted to show how a typical beginning administrator might act, and thus chose not to do these updates.

How can I protect against similar problems?

This hack, and ones similar to it, first involved a hole from the outside in, then another hole which allowed for a jump in access from limited to superuser. The recommendations below are by no means comprehensive, but are given to provide an area of emphasis to deter hacks of this type.

It is vitally important that a user keeps up to date with their distribution and applies security fixes as soon as possible. Most distributions of Linux have updates on their web site that are freely accessible. Never just set up an install from the CD and be done with it. Check for and apply the security updates. Also remove any applications from the server that aren't required. The crontab exploit that this hack used was well known at the time, and Red Hat had a patch available. Subscribe to your distributions mailing list, and they will even send you a message whenever security updates are available.

Scrutinize any CGI scripts that you plan to use. Eliminate any ones that aren't used, and any sample scripts. Check the file permissions on them to make sure they aren't writable.

Carefully look at the code involved in the CGI scripts, make sure any user changeable data has the appropriate checks on it. Anything based on HTTP variables, or dealing with user input should be considered suspect.

Make sure that the parameters for any calls to open() and system() are checked and stripped of undesired effects. Make sure that any spurious characters are eliminated from them before allowing the call to open, or system. People often neglect the backslash, and the NUL character. Without these checks, people can open files that they shouldn't be such as the /etc/passwd file, violating confidentiality. Or they could make calls that remove or change files, violating integrity and possibly availability.

The Phrack article that helped the hacker can help us as well.

In perl, to eliminate the NUL character you can:

```
$insecure_data=~s/\0//g;
```

You can then escape out the remaining shell characters with:

```
$insecure_data=~s/([\&;\'\\\|"*?~<>^\(\)\[\]\{\}\$\n\r])/\\
$1/g;
```

If you keep up with the latest security updates from your vendor, and make sure that those parameters are scrutinized, this hack can be prevented. Do not, and this must be stressed, do NOT simply assume your firewall will totally protect you.

Conclusion

In 1999 a hacker going by the name Jfs successfully answered PC WEEK'S challenge to break into their Linux box. He managed to do so by exploiting a CGI vulnerability. PC Week secured the system according to recommendations in the Linux How-TO, as well as the configuration changes on the Apache Software Foundation's Site. This is a good start in security, however it is obvious more is needed to make a box secure.

Interestingly, PC Week magazine had set up an NT server with a concurrent contest that was not hacked. This could be because they installed various service packs and security upgrades for NT, and chose not to install them for Linux. They got a lot of flack on www.slashdot.com, "News for Nerds, Stuff that Matters" for this. PC WEEK claims that NT is easier to secure and that they were put off by the 21 open-source security fixes for Red Hat 6.0 at the time. This claim holds little water though, as the security patches are all fairly small, quick to apply, and require few if any reboots, unlike NT Service packs which can take two or three system reboots each.

PC Week sums it up nicely by stating "Companies that don't keep on top of application fixes will be at the mercy of hackers who do."

References:

Jfs, "A practical vulnerability analysys"
http://hispahack.ccc.de/en/mi019en.htm
(02/01/2001)

Chowdhry, Pankaj, "CGI script opens door" (10/04/1999)
http://www.zdnet.com/eweek/stories/general/0,11011,2346293,00.html
(02/05/2001)

Chowdhry, Pankaj, "PC Week Labs' site gets hacks and flak" (09/27/1999)
http://www.zdnet.com/enterprise/stories/linux/news/0,6423,2341342,00.html
(02/05/2001)

Tascheck, John, "Hack this: PC Week Labs site begs attacks" (09/20/1999)
http://www.zdnet.com/eweek/stories/general/0,11011,2336675,00.html
(02/05/2001)

Chowdhry, Pankaj, "Attacked and hacked!" (10/11/1999)
http://www.zdnet.com/eweek/stories/general/0,11011,2350743,00.html
(02/07/2001)

rain forest puppy, "Perl CGI problems", Phrack Magazine, Vol. 9 Issue 55 p 7,
(09/09/1999)
http://phrack.infonexus.com/search.phtml?view&article=p55-7
(02/05/2001)

Last Updated: September 22nd, 2017

Upcoming Training

Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS New York SEC401^	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
SANS DFIR Prague Summit & Training 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS Phoenix-Mesa 2017	Mesa, AZ	Oct 09, 2017 - Oct 14, 2017	Live Event
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
SANS Tysons Corner Fall 2017	McLean, VA	Oct 14, 2017 - Oct 21, 2017	Live Event
SANS Tokyo Autumn 2017	Tokyo, Japan	Oct 16, 2017 - Oct 28, 2017	Live Event
CCB Private SEC401 Oct 17	Brussels, Belgium	Oct 16, 2017 - Oct 21, 2017	
SANS vLive - SEC401: Security Essentials Bootcamp Style	SEC401 - 201710,	Oct 23, 2017 - Nov 29, 2017	vLive
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC401: Security Essentials Bootcamp Style	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, United Arab Emirates	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Miami 2017	Miami, FL	Nov 06, 2017 - Nov 11, 2017	Live Event
Community SANS Vancouver SEC401^	Vancouver, BC	Nov 06, 2017 - Nov 11, 2017	Community SANS
Community SANS Colorado Springs SEC401~	Colorado Springs, CO	Nov 06, 2017 - Nov 11, 2017	Community SANS
SANS Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
SANS Sydney 2017	Sydney, Australia	Nov 13, 2017 - Nov 25, 2017	Live Event
SANS San Francisco Winter 2017	San Francisco, CA	Nov 27, 2017 - Dec 02, 2017	Live Event
SANS London November 2017	London, United Kingdom	Nov 27, 2017 - Dec 02, 2017	Live Event
Community SANS St. Louis SEC401	St Louis, MO	Nov 27, 2017 - Dec 02, 2017	Community SANS
Community SANS Portland SEC401	Portland, OR	Nov 27, 2017 - Dec 02, 2017	Community SANS
SANS Khobar 2017	Khobar, Saudi Arabia	Dec 02, 2017 - Dec 07, 2017	Live Event
SANS Munich December 2017	Munich, Germany	Dec 04, 2017 - Dec 09, 2017	Live Event
Community SANS Ottawa SEC401	Ottawa, ON	Dec 04, 2017 - Dec 09, 2017	Community SANS
SANS Austin Winter 2017	Austin, TX	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Bangalore 2017	Bangalore, India	Dec 11, 2017 - Dec 16, 2017	Live Event
SANS vLive - SEC401: Security Essentials Bootcamp Style	SEC401 - 201712,	Dec 11, 2017 - Jan 24, 2018	vLive