

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-12/msg00086.html>

- *From:* SecuriTeam <support@xxxxxxxxxxxxxx>
 - *Date:* 26 Dec 2005 18:51:08 +0200
-

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>
-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.
<http://www.securiteam.com/maillinglist.html>

Electric Sheep Screensaver Multiple Vulnerabilities

SUMMARY

" <<http://electricssheep.org/>> Electric Sheep is a free, open source screen saver run by thousands of people all over the world. "

Multiple vulnerabilities have been discovered in Electric Sheep that allows attackers to cause the program to execute arbitrary programs and code.

DETAILS

Vulnerable Systems:

- * Electric Sheep version 2.6.3

Immune Systems:

- * Electric Sheep version 2.6.4

An Electric Sheep client is downloaded by an individual who wishes to view the screen saver. Upon execution, the Electric Sheep client periodically requests a list of available sheep from the sheep server (v2d6.sheepserver.net, by default). The server responds with a list of

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

sheep that can be downloaded through the Coral Content Distribution Network. Depending on the user's preferences, mpeg files begin downloading from the site specified in the list.

Once a complete mpeg has been downloaded, it is entered into an array of valid sheep by the client. Downloaded sheep are randomly displayed from the array with a configurable number of repeats. Along with the mpegs, points describing a fractal are downloaded and used to create, frame by frame, new fractals. These frames are then uploaded to the sheep server through a CGI running on Apache. The server then recomposes mpegs from the frames given to it by all clients. These "sheep" are then entered into the available list and made available to distribution points, behind the Coral CDN. A voting mechanism allows users viewing the "sheep" to control its fitness in sheep generation.

A list of all environment variables used within the program was compiled. A manual dataflow analysis yielded a single environment variable whose handling was susceptible to memory corruption similar to command line errors. This was the HOME environment variable. There are many techniques that can be used to write successful exploits using data retrieved from environmental variables.

The HOME environment variable presents several significant barriers to exploitation. For example, changing the HOME environment variable before logging in will prevent a successful login on the OS itself. This can possibly be bypassed by sourcing new environment variables immediately before the attack and changing them back afterwards.

Since we would have to run the program from the command prompt to attack it with the environment variable, we could easily change a variable during a session by using the shell's built in command. Bash uses the command export while on t/csh it is possible to use setenv to dynamically change your HOME environment variable. These changes will not affect the default variables which are applied to a user account when he or she logs onto the system. The user may also store a saved HOME variable exploit into a batch file and use `source` to load the changes into the current shell environment.

Additional analysis was done with BFBTester, the Brute Force Binary Tester. This application compiles a list of all possible command line and environment variable inputs to a given binary and then attempts to overflow them with ever increasing accuracy.

The test was run on multiple versions of Linux. Similar tests were not available for the Windows client. The results of BFBTester being run on the Electric Sheep client are presented below:

```
$ bfbtester -a electricssheep
=> /usr/bin/electricssheep
* Single argument testing
* Multiple arguments testing
```

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

```
* Environment variable testing
*** Crash </usr/bin/electricsheep> ***
args: [10240]
envs: HOME=[10235]
Signal: 11 ( Segmentation fault )
Core? No
*** Crash </usr/bin/electricsheep> ***
args:
envs: HOME=[10235]
Signal: 11 ( Segmentation fault )
Core? No
*** Crash </usr/bin/electricsheep> ***
args: [10240]
envs: PATH=[10235]
Signal: 15 ( Terminated )
Core? No
*** Crash </usr/bin/electricsheep> ***
args:
envs: PATH=[10235]
Signal: 15 ( Terminated )
Core? No
```

The following code excerpt lists the relevant parts of code pertaining to the HOME environment variable memory corruption vulnerabilities existing in the electricsheep.c file.

```
1. HOME
electricsheep.c:
#define MAXBUF (5*MAXPATHLEN)
...
void flags_init(int *argc, char ***argv) {
...
...
if (!leave_prefix) { //default if leave_prefix not
supplied on command line
char b[MAXBUF];
char *hom = getenv("HOME");
if (!hom) {
fprintf(stderr, "HOME envar not defined\n");
cleanup_and_exit(1);
}
sprintf(b, "%s/.sheep/", hom);
leave_prefix = strdup(b);
} else if (leave_prefix[strlen(leave_prefix)-1] !=
'/') {
char b[MAXBUF];
sprintf(b, "%s/", leave_prefix);
leave_prefix = strdup(b);
}
...
if (1) {
```

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

```
char b[MAXBUF];
sprintf(b, "mkdir -p %s", leave_prefix);
mysystem(b, "mkdir leave prefix");
/*this call will fail before the function returns,
so exploit won't be executed.*/
}
...
...
return;
...
}
```

Although it was initially thought that the buffer overflow vulnerabilities in this code would be able to allow arbitrary code execution, upon manual attempts to exploit, it was found that Electric Sheep properly handles large buffers in the HOME environmental variable. The reason is outlined below.

In the electricssheep.c source code file there exists a possible buffer overflow vulnerability in the flags_init function. Even though the buffer is easily overflowable, it seems that it is not necessarily exploitable. The reason is that between the function return and the actual buffer overflow there is a piece of code that must execute. That is, the system call to the (mkdir) function must be made.

Unfortunately, if the vulnerable buffer is overflowed with more than MAXBUF (actually MAXPATHLEN) chars the mkdir function will fail. When the function fails, the program quits. Quitting the program prevents the flags_init function from returning and running any possibly injected code.

As mentioned above, dependencies can pose a threat to the system if they are exploitable. cURL is a command line tool to transfer data from or to a server, using one of the supported protocols (HTTP, HTTPS, FTP, FTPS, TFTP, GOPHER, DICT, TELNET, LDAP or FILE). The command is designed to work without user interaction. It supports NTLM authentication to retrieve files from Windows-based systems.

On October 13, 2005 a vulnerability in the cURL command was found for most Linux platforms. This vulnerability related to the use of NTLM authentication and allows a malicious attacker to whom cURL connects to, to overflow an internal buffer causing it to execute arbitrary code. The vulnerability can be exploited in Electric Sheep the following way, without changing the source code:

```
electricssheep.c
1966:
if (proxy_name) {
sprintf(curl_cmd, "nice -n %s", nice_level,
proxy_name);
if (proxy_user) {
strcat(curl_cmd, " --proxy-user ");
strcat(curl_cmd, proxy_user);
```

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

```
}
```

We can set up proxy_name variable and proxy_user variable to complete the command as follows

```
nice -n nice_level curl --proxy proxy_name --proxy-user  
(username1:password1 --proxy-ntlm -u username2:password2  
URL)
```

The variables are set up with a macro reading arguments from command line:

```
sarg("--proxy-user", proxy_user)  
and  
sarg("--proxy", proxy_name)  
which is defined as:  
#define sarg(oname, vname)  
if (!strcmp(oname, o)) {  
if (*argc > 2)  
vname = (*argv)[2];  
else goto fail;  
(*argc)--2;  
(*argv)+=2;  
}
```

Therefore, into the command line at electricssheep startup we enter something along the lines of:

```
--proxy_user username1:password1 --proxy-ntlm -u  
username2:password2 URL  
username1:password1 argument does not matter, since username2:password2  
overrides it, and URL should be a plaintext URL to the proxy server  
without quotes.
```

The input necessary to cause overflow is a user name and domain name that together are longer 192 bytes. cURL also contains another form of this vulnerability that poses a greater threat to users. The server that cURL connects to may send a redirect allowing the passing of a url greater than 192 bytes in length overflowing the buffer it is stored in which could allow the attacker to gain access to the system with the user's privileges. Visit <<http://curl.haxx.se/mail/lib-2005-10/0061.html>> <http://curl.haxx.se/mail/lib-2005-10/0061.html> for further information regarding these possible vulnerabilities.

cURL is also used in one other place in electricssheep code (line 90 electricssheep_voter.c), however this use is not exploitable.

The control information distribution process was reverse engineered through manual code analysis of the server and client and live network sniffing with Ethereal. This yielded an understanding of control information flow across the Electric Sheep system. Further analysis of this process discovered that none of the control information was checked for integrity. This was demonstrated in the initial downloading of the available sheep list.

Appended to the end of each sheep are the instructions for the creation of

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

new frames of sheep to be sent back to the sheep server.

This information is also not validated.

To demonstrate these vulnerabilities in system design, a rogue sheep server was created. Through the use of DNS poisoning, the flow of control information was redirected to the rogue sheep server. This server then supplied instructions to the client to retrieve our own forged sheep mpegs. As long as they contained a proper footer, these mpegs were entered into the array of available sheep by the client without any authentication and displayed on the clients screen. At minimum, an attacker would need to implement only two functions of the sheep server: the list of available sheep (list.cgi) and rogue sheep themselves (sheep.mpg).

The Electric Sheep project uses a program called flam3 to create fractal flames based on a genetic sequence allowing the distributed creation of sheep. This library may be susceptible to memory corruption through malformed input from the rendering of genetic sequences or local environment variables set. While only one instance was found, the authors believe more exist.

1. format

flam3-animate.c:

```
char *prefix = args("prefix", "");
```

```
char *format = getenv("format");
```

```
.....
```

```
fname = malloc(strlen(prefix) + 20);
```

```
.....
```

```
sprintf(fname, "04d.%s", prefix, ftime, format);
```

Other conditions came to light while performing repeated analysis of the code which we did not actively try to exploit but did recognize as possible security problems.

strlen has potential security implications, use strlen instead.

strcmp has potential security implications, use strncmp instead.

sprintf has potential security implications, use the following as an illustration of how to fix the problem using snprintf

definition: char pbuf[MAXBUF];

problem: sprintf(pbuf, "s %s", curlcmd, tfb,

server_anims[idx].url);

fix: snprintf(pbuf, (size_t) MAXBUF, "s %s",

curlcmd, tfb, server_anims[idx].url);

The threaded nature of the client lends itself to potential race condition attacks:

stat() and unlink() could be susceptible to TOCTOU (Time of Check, Time of Use) attacks

Race conditions are difficult to test and have minimal impact on this application.

[NEWS] Electric Sheep Screensaver Multiple Vulnerabilities

By spoofing the DNS entry for sheepserver.net or otherwise redirecting the Electric Sheep client to a malicious sheep server, it is possible to force the Electric Sheep client to download and display arbitrary mpegs due to a lack of authentication of the sheep server and sheep mpegs. At minimum, a rogue sheep server would need to respond to the Electric Sheep client with list.gz, a list of sheep available for download, and the referenced mpegs. To properly display the mpegs, they need to contain special footer information which can be found at the bottom of any pre-existing Electric Sheep mpegs.

Electric sheep uses cURL internally for interaction with the Electric Sheep server. Two recent vulnerabilities in cURL can be exploited through malicious interaction with the Electric Sheep client.

As in the previous vulnerability, spoofing the DNS entry of sheepserver.net or otherwise redirecting the Electric Sheep client to a malicious sheep server and replacing it with an appropriate HTTP 30x response can allow remote code execution through cURL due to an NTLM buffer overflow vulnerability.

Calling the Electric Sheep client by command line, configuration file, or otherwise with a malicious sheep server URL allows local code execution through cURL due to a URL buffer overflow vulnerability.

In addition, by redirecting the Electric Sheep client to a rogue sheep server and supplying a list of maliciously formatted URLs it is possible to exploit the same cURL URL buffer overflow vulnerability remotely. This is possible because the Electric Sheep client makes direct system calls to the vulnerable cURL application from network supplied input.

Spoofing the DNS entry for sheepserver.net or otherwise redirecting the Electric Sheep client to a rogue sheep server, it is possible to remotely control the video displayed or remotely execute code on all Electric Sheep clients affected by such a redirection. Local code execution is also possible due to a cURL vulnerability.

ADDITIONAL INFORMATION

The information has been provided by
<<mailto:michaelaiello@xxxxxxxxxxxxxxxxxx>> Michael Aiello.

=====

This bulletin is sent to members of the SecuriTeam mailing list.
To unsubscribe from the list, send mail with an empty subject line and body to:
list-unsubscribe@xxxxxxxxxxxxxxxxxx
In order to subscribe to the mailing list, simply forward this email to: list-subscribe@xxxxxxxxxxxxxxxxxx

=====
=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.

-
- Prev by Date: [*\[NT\] Interaction SIP Proxy Heap Corruption Vulnerability \(Long REGISTER\)*](#)
 - Next by Date: [*\[UNIX\] Linux procfs Information Disclosure*](#)
 - Previous by thread: [*\[NT\] Interaction SIP Proxy Heap Corruption Vulnerability \(Long REGISTER\)*](#)
 - Next by thread: [*\[UNIX\] Linux procfs Information Disclosure*](#)
 - Index(es):
 - ◆ [*Date*](#)
 - ◆ [*Thread*](#)