SANS ANALYST PROGRAM

Sponsored by FireEye

New Tools on the Bot War Front: FireEye 4200 Appliance Review

A SANS Whitepaper – December 2009

Written by Jerry Shenk

Enterprise Impact and Avoidance

Appliance Setup

Dashboard

How Bot Attacks Work

Lab Setup

FireEye 4200 Appliance Testing

Executive Summary

Over the past 10 years, computer attacks have shifted from operating systems to applications. On the client side, the easiest target has become the user with a web browser. Users often browse to a seemingly innocent site and get loaded up with malware that their antivirus isn't catching. The worst of this malware connects the computer to a botnet. The individual computers (or drones) on the botnet are controlled by the botnet operator. These drones can be used to scan the Internet or their local network, collect login credentials and passwords, run phishnets, instigate DoS attacks, or do a variety of other things the computer and network operator isn't aware of.

According to a recent study[1] by The Internet Storm Center and SANS Institute, browsers and their plug-in applications are increasingly being attacked by installers hidden on legitimate web sites that have been compromised by attackers. Using SQL injection, cross-site scripting and iFrame injections, criminals take over web sites and hide their Trojan horse installers. When the victims go to the site, in addition to getting the expected web content, they also get infected with a Trojan horse, which in many cases will then connect the computer to a bot network. In some cases the bots stand idle, waiting for orders. One example of this type of bot is the Confiker worm bot that is continuing to spread.

In most cases, bots are put to work quickly. By the end of this year (November 2009), volunteers at Shadowserver.org were monitoring more than 260,000 bots in operation (not resting/waiting bots) across 4,200 separate, active botnets—up from 3,000 botnets at the beginning of 2009. Botnets usually average in size from 20,000 to 40,000 drones, but some contain more than a million drones. These botnets have high value to the criminal underground because they are powerful, flexible and reusable, and they are easy to hide, command, and move around. Once the bot network has enough CPU power, it can be sold, rented or used in other takeover attacks.

Traditional security controls aren't catching these infections through the browser because they hide in HTTP traffic and take advantage of approved, legitimate browser plug-in applications. The FireEye 4200 appliance is designed to watch the network for possible malware, test it in a virtual environment, and send alerts. In case an infection has already happened, it also monitors for outbound activity and reports problems. This paper discusses the bot problem and its impact on enterprises, followed by a review of how FireEye's 4200 appliance catches bot installers and bot traffic on the network.

[1] www.sans.org/top-cyber-security-risks

Enterprise Impact and Avoidance

To an enterprise, bots create a multidimensional problem by acting as infectors on internal networks, as well as acting as thieves by stealing corporate intellectual property and personal data from network traffic and infected systems. Bots are difficult to detect because they usually enter through the browser, hide themselves from the operating system and security, and transfer data and commands over HTTP channels while looking like legitimate web traffic. They can also spread themselves further into the network, where they can be placed on sensitive systems to capture and transmit sensitive data to the bot operator.

For example, Albert Gonzalez and his cohorts are accused of breaking into the payment processing system provider, Heartland Systems, initially through a SQL injection into one of its web pages in December 2007. That breach was detected and the hole was patched, but nothing was detected inside the network. Then, in May of 2008, a sniffer activated on an internal payment system. This was a month after Heartland passed a PCI DSS audit.[2] The sniffer was not a bot, per se, because it was not part of a large remote-controlled network. However, this example does show how difficult remote-controlled malware is to detect once it gets inside the network and how easily it can hide command/control and outbound data transfers in HTTP.

On an organizational level, protection starts at web sites to ensure that they cannot be taken over and used to install bot controllers on visiting browsers. So, organizations with public-facing web sites must include assessment, monitoring, strong authentication and secure application coding in their processes.

For the endpoint, the most critical protection is training end users about safe Internet use—avoiding questionable sites, including file-sharing sites and questionable links; keeping browsers patched and up-to-date, and so on. Nowadays, most users know not to click unsolicited links in e-mail messages, and they try to navigate sites safely. However, criminals find new ways to outsmart users. It's often hard to tell, for example, when criminal ads have been posted on legitimate sites—ads that can infect a computer that simply browses the ads. Some of the base content (not just ads) of legitimate sites has even been infected.

In addition to user awareness, the Shadowserver Foundation, a group of security professionals that monitors the dark side of the Internet for bots and bot controllers, advises keeping workstations patched and updated as another key protection.[3] In particular, this means maintaining the browser and browser application plug-ins such as Adobe Flash and others.

[2] www.justice.gov/usao/nj/press/press/files/pdffiles/GonzIndictment.pdf
[3] www.shadowserver.org/wiki/pmwiki.php/Information/Botnets

In an enterprise, one place to begin getting control of bot infections is on the network. It's important to monitor for implications of botnet activity at ingress and egress points that indicate command and control or drone activity. This traffic is increasingly being conducted over HTTP, according to Shadowserver, which means monitoring needs to look deep into HTTP traffic. You must monitor both inbound and outbound traffic to detect the initial attack and the subsequent botnet activity.

To this degree, the FireEye 4200 appliance monitors web traffic and runs any programs that looks suspicious inside its virtual machines (VMs) for further testing. These VMs run the Windows XP operating system and have a variety of common business applications installed. Knowledge of botnet activity is coordinated through the FireEye Malware Analysis & Exchange (MAX) Network, which is regularly updated with bot information. Together they make up the FireEye Analysis & Control Technology (FACT) engine, which monitors the VMs and generates logs and sends e-mail alerts when bots are suspected. The 4200 dashboard has an Alerts tab that summarizes attack activities and provides access to details—down to decoded packet captures or raw packet captures (pcap) that can be used to locate bot-infected devices.

By networking with other monitoring/security appliances, and with continual analysis by the FireEye security team, the 4200 can help organizations stay up-to-date with the ever-changing botnet landscape.

Appliance Setup

The FireEye 4200 appliance is easy to set up. It fits in one unit of rack space and extends about 20 inches into the rack, so it's neither a huge box nor a tiny device. The interfaces are all clearly marked and the four-page Quick Start guide was enough to get me up and running with the system. After connecting the power cable and two Ethernet cables, I used the included serial cable to connect to a laptop. I then used my terminal emulation software to go through the 13-step setup. I used Minicom, an application on the Linux notebook that was next to the 4200 in the rack. I also tested Hyperterminal from a Windows computer, and that worked just as well. The setup survey asked for a hostname, IP address, routing information and connections to the FireEye Malware Analysis & Exchange (MAX) Network. Once the basic setup was complete, I connected to the Graphical User Interface (GUI) from a browser and logged in with the credentials that I had just set up.

The instructions recommend using Interface 1 for management and giving that an address on the network. The IP address of 10.1.1.199 was assigned for the lab network. I then connected the first of the four monitoring interfaces to the port on the lab's switch that mirrors all Internet traffic. This was enough to have things operational. The graph on the FireEye Dashboard immediately showed the total traffic and the HTTP (web) traffic. Because this appliance is specifically configured to monitor web traffic, this gives a quick "sanity check" of recent traffic to verify that it is, indeed, seeing traffic.

The area that will probably cause most problems during installation is the switch configuration. The lab has a combination of Cisco, Dell, HP, Cabletron and other switches that configure their monitoring or port redirection features differently. On the lab's core switch, Port 1 was redirected to Port 7. Port 7 was then connected to Interface 3 (the first monitoring port) on the FireEye appliance.

I connected the second monitoring interface (Interface 4) to an internal switch that is connected to the "victim computers" in the lab. These victim computers are computers that were intentionally compromised during testing. Many environments initially use the first monitoring interface, but it's good to have additional ports available for a server network, a DMZ, or for testing.

The FireEye 4200 has an option for sending alerts via e-mail, which is often the best way to receive notification that a machine is actually infected, especially if it actually starts participating in a botnet. Given the fact that one feature of the appliance is the virtual elimination of false positives, I wanted to use this. Unfortunately, when I set it up from the GUI, it didn't work right. I called tech support and found that I needed to go to the command-line on the appliance and set e-mail notification up manually. They walked me through that process, and once it was correctly configured, e-mail notification worked flawlessly. The reports have the most pertinent information in the first five lines, with details included in machine-readable XML for automated processing, but they are also in human-readable plain text.

That's really all the setup there was. It was quite straightforward, and it just worked. Periodically, I went into the Appliance Settings tab of the GUI and then to the MAX Network tab to verify that the appliance was getting updates—and it was. The appliance gets updates every few days.

Dashboard

The appliance has a web-based dashboard that serves as a constant status indicator of what's going on with the network. While I was testing this appliance I liked having the dashboard displaying that nothing was going on. For an appliance like this, nothing going on is good. Figure 1 is a screenshot I took while actively infecting the "victim" a number of times one afternoon. If this were displayed on a large monitor hanging on the wall in the computer support center, NOC, SOC or something similar, this would be a good indication that somebody should be looking into the infected clients. It is better to catch the infection now before just one infected machine starts infecting its neighbors. The details of the infected clients are recorded under the Alerts tab and include detailed packet analysis. These features make it easy to locate infected machines.

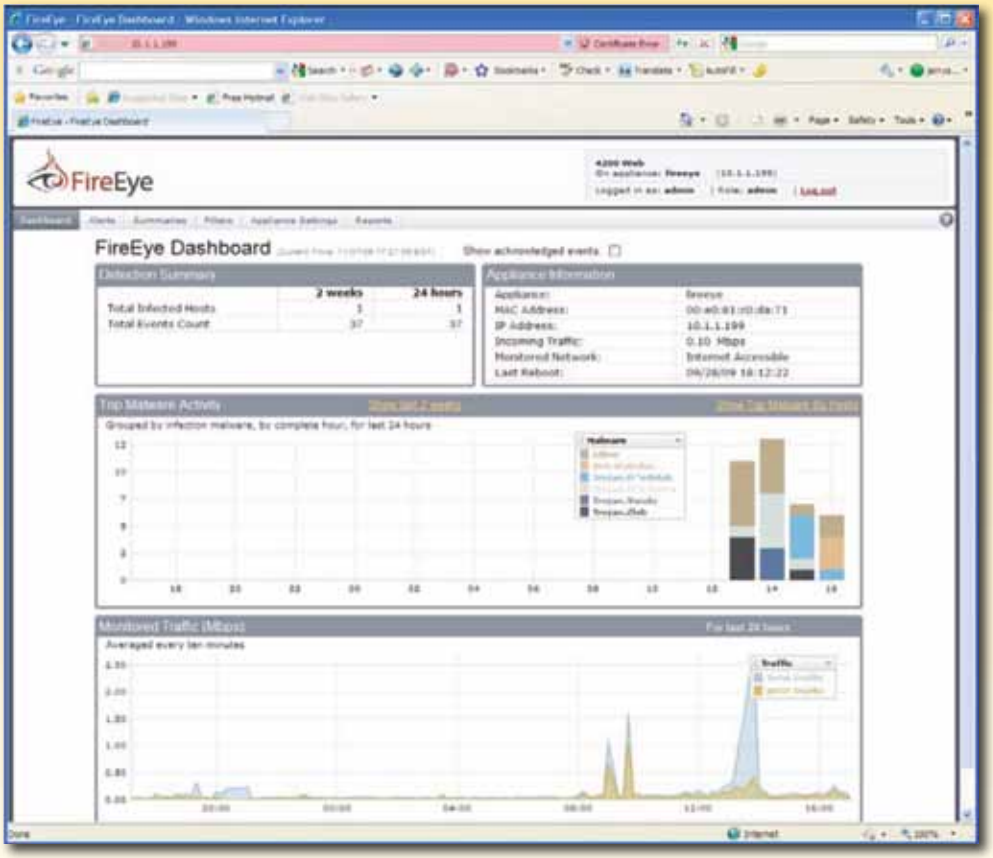

Figure 1: The 4200 Dashboard Catching a Single Machine Infection

By default, the GUI will log out the monitoring account after 10 minutes of inactivity. When installing the FireEye 4200 in a NOC or other limited access facility, the default inactivity timeout can be disabled from the command line by using an SSH terminal program to log in using the admin credentials. The command line is similar to a Cisco router command line. At the **config** prompt, a command of **webgui** auto-logout 10080" would keep the screen active for a week.

How Bot Attacks Work

When deploying any security solution, it is important to understand exactly how it fits into an overall security plan. FireEye doesn't claim that this appliance is a silver bullet that will solve all your security problems. (In fact, if you find something that's advertised to solve all your problems, be very skeptical.) The FireEye 4200 is specifically designed to monitor and analyze web traffic, specifically traffic that would be dangerous to computers running commonly attacked software on the Windows XP operating system, which is the most prevalent and attacked desktop operating system in use today.

Most bot traffic starts off with a seemingly innocent link, web advertisement or an e-mail ad. Porn and free software seem to be quite popular sources of infection, but legitimate sites often get hacked to include hostile code that is loaded onto visiting browsers. Some popular carriers are ads for AntiVirus Pro 2010, Cyber Security, and Free Online Scanner. Once the users click on the link to install the software.

After the malware is installed, the victim computer joins the botnet and starts doing whatever is asked of it. The term botnet relates to a collection of drone or zombie computers with bots running on them. These bots, in turn, are controlled by the bot operator. In the first infection of our victim machine in the lab, the initial thing the infected system did was connect to a server in Russia (.ru domain) and request some work to do:

```
GET
/new/controller.php?action=bot&entity_list=1241530597,1255137485,125549
2056,1255594149&uid=1&first=0&guid=3695208254&rnd=981633 HTTP/1.1
```

This seems very blatant, not even trying to hide at all—**action=bot**. That indicates that they don't care too much about getting caught. There are so many victims that the bot's owners are more interested in growing their bot armies than worrying about one drone getting caught. In this particular example, there was some additional web traffic similar to the above **GET** example. Then, the victim machine started attempting to send e-mail messages (most were failing to be delivered). While sending e-mail messages, another executable program (**install.exe**) was downloaded. A very quick analysis of **install.exe** indicates that it was another spam mailer bot.

The second bot infection was delivering small bursts of e-mail about an online pharmacy that read,

> *"Canadian pharmacy is as good as American pharmacies but offers much lower prices on medications. High quality generic medications— easy to order and delivered in discreet packages! Its absolutely secure and safe! Explore the trustful source for low-cost meds."*

Because somebody we don't know or trust controls these drones, there can be no such thing as a safe drone. Just because the computer is dedicated to spam delivery today doesn't ensure that the computer won't start scanning the internal network for more access tomorrow, as in the Heartland case.

In addition to sending spam, bots have been known to harvest personal information from the local machine, scan and attack other machines on the local network or Internet, share CPU cycles for computation-intensive tasks, participate in Distributed Denial of Service attacks (DDOS), and other uses.

These crimes ends up costing organizations time and money to locate and remedy the bot infections and downstream victims. There is also damage to the reputation of organizations that discover bots in their networks or end points that have been harvesting personal information on their employees and/or their customers.

Lab Setup

I tested the FireEye 4200 appliance in a lab designed to ensure that that the bot traffic did not affect the rest of the network. The lab network has a quarantine network within it that is only allowed to get to the Internet and the lab web/DNS server. This restriction was created with a router access control list (ACL). This allows captured malware to be staged on the lab web server where it can be downloaded by the intended victim, while the other computers in the lab are protected from attack by these victims. These victim computers (drones) are configured to use a lab DNS server so that DNS requests can be monitored. The 4200 appliance was positioned where it would see all Internet traffic on one interface and all traffic from the quarantine network on another interface. Both interfaces were needed in order to stage some malware on the lab web server and to grab traffic in the wild.

The main "victim computer" was a Windows XP workstation with Service Pack 2 and Adobe Reader version 7.05. For this testing, I used hardware computers instead of virtual computers. Some malware will detect that it's running in a virtual environment and shut down because virtual environments are often used in malware testing. To avoid that problem, I loaded the operating system on a 6 gigabyte partition of a drive with a larger partition formatted in a Linux format to be used for archiving. The XP partition could then be imaged in its uninfected state and recovered after the infection.

Image Creation/Recovery

To create and recover the image, the Helix 1.7 CD was used to boot the computer. Most versions of Linux have *dd* and *gzip* and would work similarly. This particular computer is a retired server and has a single SCSI drive. The device names are peculiar to each system. The dd program was used for the creation and restoration process, and the image was compressed to allow multiple images to be stored.

Create Original Image

```
mkdir /mnt/sda3

mount /dev/sda3 /mnt/
sda3

dd if=/dev/sda2 bs=2k |
gzip > /mnt/sda3/XPPRO_
SP1.img.gz
```

Restore Image After Infection

```
mkdir /mnt/sda3

mount /dev/sda3 /mnt/
sda3

gzip –dc /mnt/sda3/
XPPRO_SP1.img.gz | dd
of=/dev/sda2
```

[4] www.thepiratebay.org

There were three malware executables staged on the internal web server. The first staged malware was loaded using Internet Explorer version 6 by browsing to a specific page on the lab web server. The other malware was loaded the same way.

The malware in the wild was loaded by going to the Pirate Bay file-sharing site.[4] I looked at the top 100 hits and grabbed a few that looked interesting. It took four downloads until the machine began to be infected.

Quarantine ACL

For this testing, the quarantine lab traffic must be restricted so that it cannot affect the rest of the lab. I accomplished this with a simple ACL defining that quarantine lab traffic is permitted to go to one server in the lab's server network. Traffic is also explicitly denied any access to the networks that have been reserved in RFC 1918[5] for private traffic (10/8, 192.168/16 and 172.16/12). Traffic that is blocked is also logged so that any scanning attempts of RFC 1918 traffic can be detected. For this lab test, the quarantine lab machines are all on the 10.1.2.0/24 network with a DNS server of 10.1.1.1.

no access-list 170

```
access-list 170 permit tcp 10.1.2.0
0.0.0.255 host 10.1.1.1 eq www
access-list 170 permit udp 10.1.2.0
0.0.0.255 host 10.1.1.1 eq domain
access-list 170 permit tcp 10.1.2.0
0.0.0.255 host 10.1.1.1 eq domain
access-list 170 deny   ip 10.1.2.0
0.0.0.255 10.0.0.0 0.255.255.255 log
access-list 170 deny   ip 10.1.2.0
0.0.0.255 192.168.0.0 0.0.255.255 log
access-list 170 deny ip 10.1.2.0
0.0.0.255 172.16.0.0 0.15.255.255 log
access-list 170 permit ip any any
```

Restore Image After Infection

```
description Quarantine lab connection
ip address 10.1.2.212 255.255.255.0
ip access-group 170 in
```

[5] www.apps.ietf.org/rfc/rfc1918.html

FireEye 4200 Appliance Testing

I tested the 4200 appliance by positioning it on the network where network traffic passed its monitoring interfaces. The appliance was set up to monitor all Internet-based traffic and also to monitor all traffic on the quarantine network.

Initially, exploit code was staged on the lab web server that would be common in a penetration test. The exploits were performed using standard CORE Impact and Metasploit. This code compromised the computer but did not set off the expected alarms. FireEye tech support assisted with a number of tests to verify that the appliance was seeing the traffic. I then captured the exploit traffic on the appliance and sent it to FireEye. Tech support explained how these exploits were different from the typical botnet traffic and that prevented them from being analyzed.

The next test was to stage some malware that had been captured from the Internet using the lab's web server. Those exploits caused the victim computer to be infected with a variety of Trojan programs and then connect to a variety of botnets. This traffic was detected by the 4200 and generated the expected e-mail alerts.

Staging malware is great for verifying proper operation, but I also wanted to capture live infections of the victim computer. In this case I had the victim computer installed in a quarantine network where it was only able to harm itself.

The Pirate Bay web site is known for hosting pirated software, which is often infected with malware before being distributed. Our experience supported that idea. After downloading four files, the victim computer was quite infected, and after a review of the timeline, the probable culprit was a "registered" copy of WinRar, a popular file compression utility. When the WinRar download was executed, the FireEye appliance started sending e-mail alerts about the infections and the subsequent botnet connection attempts even before the installation was really started. Because WinRar is a self-extracting file, it is an ideal candidate for carrying a Trojan.

During less than 45 minutes on Pirate Bay, the victim machine encountered 11 different Trojans and joined six different botnets. The FireEye appliance dutifully sent out e-mail alerts as expected, and it was easy enough to locate infected machines. These e-mail messages included the IP address of the compromised computer and details about what type of malware had been installed. The details are also available from the GUI.

As stated, this tool is focused on botnet activity as it pertains to HTTP traffic traveling in and out of the network, making the appliance a useful tool in the botnet battle. However, it is just one piece of puzzle which should also include ingress and egress firewall rules, antivirus software and good patch management.

Summary

Botnets are a problem for everybody using the Internet today. Businesses must clean up their systems and networks after users stumble across infected web sites. If the organization is lucky, the infection only affects a single workstation. However, even a single infection is costly if the bot goes unnoticed and is used to mine confidential customer and/or employee data and send it out to a bot controller.

Even if you've never gotten a bot installed on your computers, you've certainly received spam telling you that a Nigerian bank has millions of dollars waiting for you or that medication for any ailment you have is only a click away. Those messages probably came from a botnet. And clicking them would likely lead to a bot infection.

Training users in safe Internet behavior is a great start. Keeping the operating system and software up-to-date is also critical—particularly web browser and browser plug-in applications, which are the primary vectors of infection today.

At the network layer, using egress filtering can also be a big help in keeping bot infections to a minimum. Since the most important part of any security program is knowing when you have a problem, the FireEye 4200 can go a long way toward notifying you quickly if machines on your network have become infected. Prevention is the goal, but detection is critical!

About the Author

Jerry Shenk currently serves as Senior Analyst for the SANS Institute and is the Senior Security Analyst for D&E Communications in Ephrata, PA. Since 1984, he has consulted with companies and financial and educational institutions on issues of network design, security, forensic analysis and penetration testing. His experience spans small home-office systems to global networks. Along with some vendor-specific certifications, Jerry holds five GIAC GOLD certifications: GCIA, GCIH, GCFW, GSNA and GCFA, all completed with honors.

SANS would like to thank this paper's sponsor: