

# Botnets: Big and Bigger

**T**his issue introduces the first of a regular series of articles based on the Honeynet Project and Honeynet Research Alliance's research. As Honeynet Project director Lance Spitzner explained in the March/April issue of *IEEE Security & Privacy*, researchers design

honeynet computer networks specifically to be attacked.<sup>1</sup>

The hosts that comprise a honeynet and serve as attack targets are called *honeypots*. Researchers configure them to capture a variety of useful data about computer attacks without compromising other computers. Moreover, honeynet researchers strive to implement data capture and control in such a way that intruders are unaware that their actions are being monitored.

Although honeynet technology is relatively new, it is developing rapidly. Honeynets have already proven themselves to be useful sources of information. In this article, I'll describe an attack on a honeypot that occurred in March 2003 during the onset and peak activity of several worms that targeted vulnerable hosts running Windows file sharing. We incorporated the compromised honeypot into a large botnet that attackers used to initiate distributed denial-of-service (DDoS) attacks against several Internet sites. I'll explain the structure of such botnets, their use by computer attackers, and the threat they pose to Internet sites.

## The honeynet

In March 2003, the Azusa Pacific University Honeynet Research Project deployed a honeypot on a Mi-

crosoft Windows 2000 server. We installed the operating system with default options, including a null administrative password. We didn't apply service packs or patches to the operating system, which made it vulnerable to a variety of common remote attacks, including password guessing. However, we assigned a strong password to the MS SQL server administrator account to prevent infection by the Slammer worm, which had recently compromised many vulnerable Internet hosts.

We deployed the honeynet within a class C network associated with a university lab used in undergraduate information technology courses. Figure 1 shows a simplified version of the honeynet architecture. Notice that the host running the

Snort intrusion detection system is connected via a dotted line to the network; its interface is associated with a switch monitor port that lets the host listen but not transmit. The figure doesn't show a second network used in administering the honeynet. The host running Snort has a second network interface (for managing the host and archiving log files) associated with this network.

## The attacks

Almost immediately after we deployed the server, attackers and worms scanning the honeypot's class C network address block discovered the honeypot and subjected it to a variety of probes and attacks. During the first week of observation, 171 distinct IP addresses accessed the server. Ports targeted by attacks included those shown in Table 1, but several attackers performed complete port scans of the honeypot.

In particular, the Slammer worm quickly probed and attacked the honeypot. This particular worm targets a vulnerability in Microsoft's SQL server by listening on port udp/1434. However, the strong password assigned to the SQL server ad-

BILL MCCARTY  
Azusa Pacific  
University

## Honeynet tools

**M**ost honeynet researchers appear to be operating Linux-based honeynets. Certainly, more freely available honeynet tools are designed to operate under Linux than under any other platform (see [www.honeynet.org/papers/honeynet/tools/index.html](http://www.honeynet.org/papers/honeynet/tools/index.html)).

Consequently, few tools support honeynet data capture related to Microsoft Windows hosts. We and other researchers have tools

designed for data capture in the Windows environment under development. We decided to forgo instrumentation of the honeypot that we deployed in March 2003. Instead, we relied on the ability to capture data off the wire, using the Snort intrusion detection system ([www.snort.org](http://www.snort.org)) and the Tcpdump utility ([www.tcpdump.org](http://www.tcpdump.org)).

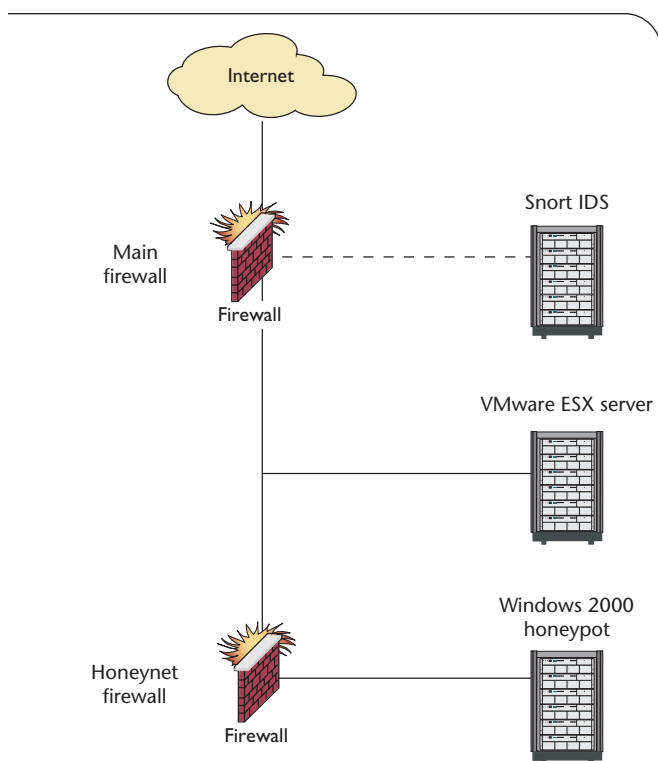


Figure 1. The honeynet architecture. The honeynet architecture supports the two essential honeynet capabilities of data capture and data control. Snort, which captures network traffic and generates alerts when attacks are detected, provides the data capture, and the dual firewalls provide data control.

Table 1. Honeypot ports probed and attacked.

| PORT     | PRINCIPAL ASSOCIATED SERVICE |
|----------|------------------------------|
| tcp/80   | HTTP                         |
| tcp/139  | CIFS                         |
| tcp/445  | CIFS                         |
| udp/137  | NetBIOS                      |
| udp/1434 | MS SQL server                |

ministrator account thwarted the worm's attack. Of course, we could have simply blocked port udp/1434 at the honeynet firewall, but our hope was that a new attack might present itself, so we left it open. Fortunately, we weren't significantly at risk of becoming an active carrier of an improved version of Slammer had one appeared. Rate limiting imple-

mented in the honeynet firewall, which constrains the number of outbound packets that can be sent in a fixed period of time, would have prevented the honeypot from attacking and compromising more than a handful of vulnerable hosts before manual intervention. Because the honeypot wasn't vulnerable to Slammer, the rate limiting features weren't triggered during operation.

During the observation period, hackers mounted several attacks (including Code Red II and a buffer overflow attack) against the Internet Information Server service listening on port tcp/80; none of these attacks succeeded. Although several attacks targeting the NetBIOS service listening on port udp/137 obtained information about users and the domain, the attacks did not lead to privileged access or access to files and folders.

Although several attacks on the CIFS service listening on port tcp/139 failed to access the IPC\$ share, seven attacks on the CIFS service listening on port tcp/445 successfully gained write access to files and folders. These attacks were also successful in escalating privileges to the administrator level, owing to the easily guessable null administrator account password. Two of these attacks involved uploading files that further compromised the honeypot's operation.

We attributed all seven attacks to worms, based on the brief time intervals (measured in seconds) over which they occurred. The attacks could have been the result of human operation of one or more mass rooters—tools designed to automatically compromise a victim host. However, no attacker took immediate possession of the honeypot, as generally occurs following compromise by a mass rooter.

We didn't foresee how much the multiple compromises would complicate post-attack analysis of the honeypot itself. In our previous experience, we saw attackers regularly close—or at least attempt to close—the vulnerabilities they had used to gain access to the honeypot. They do

this to secure their prize against attack and compromise by rival attackers. Unlike human attackers, though, many worms do not close the vulnerability by which they gain access.

## The botnet

The interesting aspect of the attack on our honeynet was not the vulnerability the attack exploited, but the actions one of the attacking worms took. The worm installed an Internet Relay Chat (IRC) client and joined the host to a nonpublic IRC network's channel, apparently via compromised hosts. At the time the honeypot was joined to the IRC network, the IRC server reported that IRC clients on 4,752 hosts were present in the channel. Inspection of network traffic showed that 15,164 distinct hosts—identified with distinct IP addresses or host names—joined the IRC channel over a period of about 10 days.

Such a structure, consisting of compromised hosts joined into a network via IRC is called a *botnet*. Some researchers use the term to refer more generally to any network formed by IRC clients not under direct human control. Such clients are commonly referred to as *bots*.

Botnets can serve several purposes, both legitimate and illegitimate. One legitimate purpose is to support the operation of IRC channels by conferring special administrative privileges on designated users. However, such legitimate purposes do not require the vast number of bots we observed via our honeypot.

Bots and botnets serve two principal illegitimate purposes: attacking IRC servers and attacking Internet sites. Many IRC networks confer ownership of an IRC channel on the first user to enter the channel; this person is known as the *founder*. The founder can, in turn, confer special privileges on other users. If all the users can be forced to exit a channel, the first user to subsequently enter it becomes the new founder. This affords the means for one or more attacking

users to wrest control of an IRC channel from its founder and the founder's delegates.

To take over an IRC channel, attackers conduct a DoS attack against one or more of the network's servers. If they can succeed in downing a server, they can split the network into two or more disconnected segments. If, in a given segment, no users are joined to a particular channel of interest, an attacker can join that channel and seize the founder's privileges.

Such juvenile antics might seem like mere inconveniences to Internet users who aren't using the IRC network and channels under attack. However, DoS attacks associated with such IRC wars can hinder the performance of sites near the attack's source or target. Would-be attackers can fill their armies' ranks by attacking and compromising any hosts belonging to third parties who don't use IRC and might even be ignorant of its existence. These innocent third parties suffer collateral damage in an otherwise invisible war.

Apart from their role in attacking IRC servers, attackers can also use botnets to attack Internet sites. They can be quite effective in such a role because the aggregate bandwidth associated with bots can be enormous. Assuming that the 15,164 bots in our observed botnet each had an associated bandwidth of 56 kbps, a simultaneous attack by the entire botnet would direct almost 850 Mbps at its target—enough to cripple any e-commerce site. We consider this estimate to be conservative because many of the compromised hosts appeared to have been cable modem and DSL hosts, which generally have somewhat larger outbound bandwidth. Moreover, because bots are widely distributed within the IP address space, filtering or blocking such DDoS attacks is not easy. At best, it requires cooperation between the target and multiple Internet service providers.

Our honeypot did not participate in DDoS attacks to any significant degree, owing to the honeynet fire-

## Virtual honeynets

We implemented the honeypot discussed in the main text as a virtual host running under VMware ESX, a commercial product that provides a virtual environment supporting several popular operating systems. Virtual honeypots are a popular way for researchers with limited access to hardware to explore honeypot technology. (Several whitepapers on the Honeynet Project Web site provide an introduction to virtual honeypots; see [www.honey.net.org/papers/](http://www.honey.net.org/papers/)) Relatively few researchers use VMware ESX as a means of hosting virtual honeynets. More popular means include VMware Workstation, VMware GSX, and the open-source User-Mode Linux. Some Honeynet Project whitepapers describe these products and their use in honeynet deployment.

Implementing honeypots as virtual hosts can provide several advantages, such as

- simple remote administration via virtual console access,

- the ability to throttle access to CPU cycles, memory, and bandwidth,
- the ability to covertly access the honeypot's file systems during operation, and
- simple set up without the need to cable up and rack the host.

However, implementing a virtual honeypot might also pose disadvantages—in particular, disguising the existence of virtualization is difficult. If an attacker discovers that a compromised host is virtual rather than real, he or she might become suspicious and abandon the host.

Nevertheless, the Azusa Pacific University Honeynet Research Project has chosen to primarily deploy virtual honeypots. In dozens of observed compromises, we have found no evidence that an intruder has checked for, and then become aware of, the existence of virtualization. We feel that the advantages of virtual honeypots could outweigh the risk of detection, at least for some sorts of honeynets.

wall's rate-limiting feature. Once we determined that it was participating in one, though, we manually blocked certain outbound ports associated with the honeypot to blunt its contribution to the overall attack.

Within a few days of the compromise that incorporated our honeypot into a botnet, another attacker compromised the honeypot, severing its connection to the botnet. Because we had captured the IP addresses of the IRC servers, the IRC channel names, and other information necessary to access the botnet, we could have insinuated a decoy to continue to monitor its operation. However, we weren't prepared at the time to do so. We weren't then—and aren't now—certain of the legal implications of such an action, which might violate privacy rights under US law.

Honeynets are a useful tool for learning about computer intruders' tools, tactics, and motives.

Although few tools are publicly available to support the deployment of Microsoft Windows honeypots, data captured off the wire reveals much about computer attacks. DDoS attacks launched by botnets, such as those we observed via our honeypot, can threaten even large and well-defended Internet sites. But because there is currently no US case law dealing with honeynet operation, US honeynet researchers should be cautious to avoid violating intruder and third-party privacy rights. Researchers who work outside the US might enjoy greater or lesser latitude.

An open area of honeynet research is the design of a reactive firewall or other mechanism that would prevent multiple compromises of a honeypot. A reactive firewall might operate by blocking inbound attacks on a particular port after detecting that the port was successfully used to compromise the host. An important general characteristic of such a mechanism is that it should operate covertly to avoid

### Honeynet ethics

Many ethical issues arise during a honeynet's operation. One of the most common questions is whether, and how, to notify network administrators of compromised hosts. The Honeynet Research Alliance's policy is to notify the Computer Emergency Response Team (CERT) at the Software Engineering Institute, Carnegie Mellon University, about any hosts compromised by an intruder attacking from a compromised honeypot. Such events are uncommon, due to the data control facilities implemented in a properly designed honeynet and the vigilance of diligent honeynet operators, who seek to prevent intruders from using compromised honeypots to harm others.

When a honeypot compromises a host, we've learned that it's generally better to notify a credible third party, such as CERT, rather than to contact the administrator of the compromised host directly. In the past, misunderstandings have arisen when network

administrators wrongly accused honeynet operators of being the instigators of attacks. Resolving such misunderstandings can be difficult and time-consuming.

The question of notifying administrators of hosts suspected of being compromised, but not compromised via a honeynet, is less clear-cut. Providing such notification might tip off the intruder and thereby prompt him or her to abandon use of the honeypot, which presents an undesired outcome from the researcher's standpoint. Moreover, a honeynet operator might lack the resources to identify and contact large numbers of administrators, as was the case in the incident described in the main article. In each such case, the honeynet operator must decide, attempting to balance research interests against the potential good that might result from notification. Advice from fellow honeynet operators is invaluable in identifying and considering the various decision factors and then coming to an appropriate decision.

raising a human attacker's suspicions.

A mechanism designed to protect honeypots from multiple compromises by worms might operate less covert because worms are likely to detect its presence. These mechanisms should block traffic only selectively because many computer intruders download toolkits in a compromise's immediate aftermath. Such toolkits are important artifacts that honeynet researchers can study. Therefore intruders' access to them should not be prevented. □

#### Reference

1. L. Spitzner, "The Honeynet Project: Trapping the Hackers," *IEEE Security & Privacy*, vol. 1, no. 2, 2003, pp. 15–23.

*Bill McCarty is an associate professor of Web and information technology at Azusa Pacific University, where he also directs the Azusa Pacific University Honeynet Project. His research interests include the use of IRC by honeynet intruders and virtual honeynets. He has a PhD in the management of information systems from the Claremont Graduate University and a BS in computer science from California State University.*

### Computing in Science & Engineering

*Computing in Science & Engineering* presents scientific and computational contributions in a clear and accessible format.

<http://computer.org/cise/>

#### July/August: **Computational Chemistry**

Computers have always played an important role in the modeling and simulation of chemical reactions. In this issue, recent computational advances in molecular dynamics, quantum chemistry, and quantum scattering will be discussed.

#### September/October: **High-Dimensional Data II**

We address challenges in handling and understanding high-dimensional data. In order to bring together different research and applications in processing and visualizing high-dimensional data to foster more insight into this fast-growing community.

#### November/December: **Optics**

This issue will explore the role of computing in the diverse field of optics. Articles will focus on subjects including modeling of optical systems and phenomena, adaptive optics in modern astronomy, and optical bio-imaging. These articles will emphasize many of the current computing challenges in optics.

