Hal Berghel and David Hoelzer

Pernicious Ports

Identifying and defending against port-related vulnerabilities.

It is now well documented that some of the greatest security vulnerabilities to computer software are a result of inattention to current service packs, hot fixes, and patches. Our experience with big-ticket malware such as W32/Blaster and SoBig illustrates the point: in both cases patches that would have prevented the infections were available at least a month before the malware was deployed (see the December 2003 "Digital Village" column). The successes of these exploits were a consequence of the millions of computers that hadn't been patched, either through neglect or ignorance.

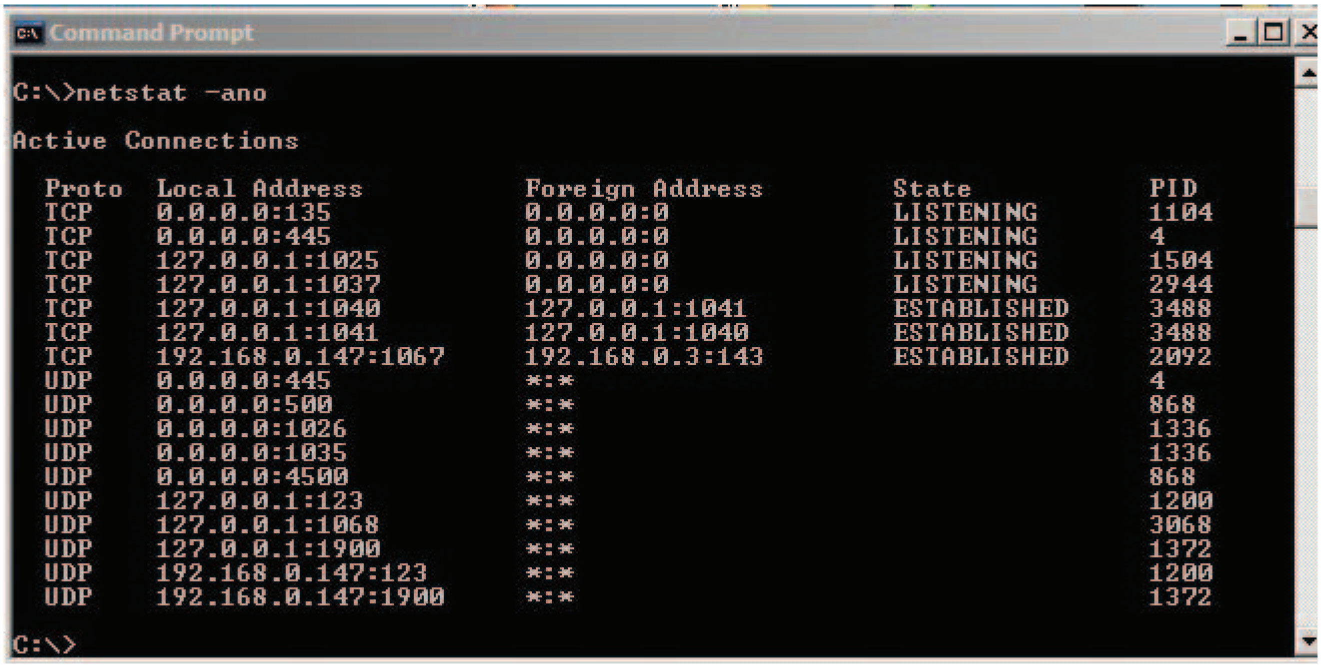
This leads us to study the art of preemption: taking preventative measures against potential insecurities. This is indeed an art rather than a science, for the problem space is enormous—the Cartesian cross product of: 131,072 available ports (65,536 each for TCP and UDP); hundreds of well-known network services (telnet, ftp); thousands of network applications (both standard and non-standard); and an ever-increasing fleet of network-mobile trojans, viruses, worms, and rootkits.

Even this analysis is oversimplified. Things are actually much worse when one adds in all of the present and future malware for which no betraying signature or operational characteristics have as yet been discovered. This problem deserves consideration regardless of our network security precautions (such as firewalls), which has been well demonstrated by worms like SoBig, NetSky, and Bagel. This column will focus on the Internet ports part of the problem space.

TCP/IP and Ports

The Internet operates on the 65,536 TCP and 65,536 UDP ports. The official IANA (www.iana.org) classification divides these ports into three categories: well-known ports (0–1023), registered ports (1024–49151), and dynamic and/or private ports (49152–65535), with many pockets of reserved ports dispersed throughout. In general, port assignments are a result of an approval process between a petitioner and IANA. However, some caveats are in order:

SANDY WONG



```
Command Prompt

C:\>netstat -ano

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       1104
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    127.0.0.1:1025         0.0.0.0:0              LISTENING       1504
  TCP    127.0.0.1:1037         0.0.0.0:0              LISTENING       2944
  TCP    127.0.0.1:1040         127.0.0.1:1041         ESTABLISHED     3488
  TCP    127.0.0.1:1041         127.0.0.1:1040         ESTABLISHED     3488
  TCP    192.168.0.147:1067     192.168.0.3:143        ESTABLISHED     2092
  UDP    0.0.0.0:445            *:*                                    4
  UDP    0.0.0.0:500            *:*                                    868
  UDP    0.0.0.0:1026           *:*                                    1336
  UDP    0.0.0.0:1035           *:*                                    1336
  UDP    0.0.0.0:4500           *:*                                    868
  UDP    127.0.0.1:123          *:*                                    1200
  UDP    127.0.0.1:1068         *:*                                    3068
  UDP    127.0.0.1:1900         *:*                                    1372
  UDP    192.168.0.147:123      *:*                                    1200
  UDP    192.168.0.147:1900     *:*                                    1372

C:\>
```

Figure 1. A typical Windows Netstat report of active ports and services.

1) IANA approves ports based on the application and intended use. There is no warranty expressed or implied by IANA that using a port is a good idea: "Assignment of a port number does not in any way imply an endorsement of an application or product, and the fact that network traffic is flowing to or from a registered port does not mean it is 'good' traffic" (www.iana.org/assignments/port-numbers). Caveat emptor is the operational metaphor. An example of an ill-advised port might be 69 (trivial file transfer protocol). This is an exemplar of high-risk internetworking because of the total absence of security features.

2) There is little control over the actual use once the port is registered. The fact that IANA approved a port neither implies that is the only use to which the port is put or that the port may be safely used. For example, IANA approved port 1999 as a Cisco identification port. However, in practice it is one of the ports used by the infamous SubSeven backdoor and Trojan.

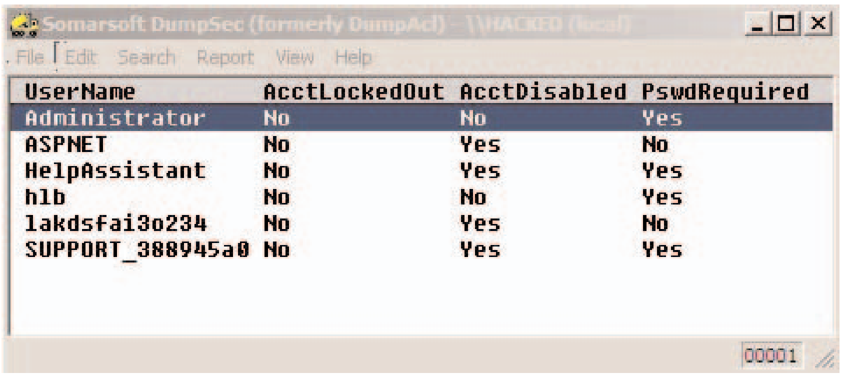
3) There is also little control over the use of ports that aren't registered. In addition to registered ports 1999 and 2773 (RBackup), for example, SubSeven also uses unassigned ports (7215 and 27374), and at least one port in the dynamic/private range (54283).

The relevance of IANA registration has declined over time as more Internet vendors wandered off on their own. Any rationale there might be for port assignments is not obvious. (Recall that the Internet was built without consideration to security and authentication.)

Port Pestilence

For convenience, we group common port vulnerabilities as Windows-centric, Unix-centric, and operating system-independent, with considerable overlap recognized. Here, we describe a few of the better-known port-related vulnerabilities you're likely to encounter.

Windows-centric. Microsoft in general, and the Windows OS in particular, receives continuous media criticism over security vulnerabilities—some deserved, some not. There is so much hostility in some of these attacks that it is sometimes difficult to remember that most of the problems result from Microsoft's commitment for maximum usability of its prod-



```
UserName          AcctLockedOut AcctDisabled PswdRequired
Administrator     No            No           Yes
ASPNET            No            Yes          No
HelpAssistant     No            Yes          Yes
hlb               No            No           Yes
lakdsfai3o234     No            Yes          No
SUPPORT_388945a0  No            Yes          Yes
```

Figure 2. DumpSec's enumerations of user account properties.

ucts. The trade-off between usability and security appears whenever a software company seeks hegemony in its market. Nowhere was this more evident than in pre-XP Windows releases that came with virtually every feature enabled.

To illustrate, from a Windows XP command prompt enter <netstat –ano>. If one combines this information that the Windows Task Manager (TASKMAN: view>select columns>check "PID">OK) provides concerning the process id, one gets a pretty good idea of the active ports and services. For example, in Figure 1 process 1504, which TASKMAN identifies as <alg.exe>, is listening on TCP port 1025 via this computer's loopback address. This raises questions such as: exactly what does <alg.exe> do?; and why does it need an open TCP connection to do it? This illustrates one of the greater security issues in Windows, namely most users are unaware that a large number of services are running, much less *why*.

The Microsoft Vista OS currently in Beta testing and scheduled for release in late 2006 will address this issue by taking a Unix-like load-time minimalism. But at least for the moment, most of the responsibility for blocking unwanted services and Internet accesses accrues to the user. Toward that end, we need some sort of "best practices" advice on which ports to close and which to leave open. This section offers a modest beginning.

TCP and UDP ports 135, 137–139 and 445 are particularly problematic in Windows. Technically, ports 135 and 137–139 are used by the legacy NetBios API while port 445 is assigned to a network protocol called Server Message Block. While originally intended for file and printer sharing among a cluster of computers in small workgroups, both may be deployed over TCP/IP as a distributed network file system. NetBios and SMB didn't scale to Internet usage gracefully. Microsoft used NetBios and SMB to support interprocess communication between and among the "network neighborhood" resources by using the SYSTEM ID. Since one can't log on as SYSTEM as it has no username or password the communication avoids both—hence the term "null session." The command to establish a null session on a local machine confirms the null usernames and passwords: net use \\127.0.0.1\IPC$ ""/user: "".

Null sessions have issues. For one, a fairly substantial body of malware uses these ports.[1] For another, there are freely available tools available that exploit null sessions. Enum (www.bindview.com) is one such utility. Enum establishes null sessions across networks and enumerates shares. It can also integrate with a lexicon to run a brute force attack on passwords and usernames. Another utility, DumpSec (www.systemtools.com/somarsoft) is a GUI-based resource enumerator. Figure 2 illustrates how simply DumpSec may be used to enumerate properties of user accounts. Beyond null session vulnerability, NetBios and/or SMB are also susceptible to a critical remote code execution vulnerability and a License Logging Service Overflow.

While NetBios and SMB can be shut off in the Windows Registry, and can be unbound from the network interface as well, it's wisest to also block the ports at the firewall as an additional precaution. Bear in mind the greatest vulnerability of NetBios and SMB is through Internet access, not from controlled internal LANs.

[1]For example, Nimda, W32.Blaster, W32/Lovsan.worm, Bugbear, Chode, Qaz on the NetBios ports; and Nimda, Backdoor.rtkit.b, Sasser, W32.HLLW.Deloder, and several W32.spybot variations on SMB, to name but a few.

Further, these Remote Procedure Call vulnerabilities now extend beyond port 135. Since RPC is set up to bind to the first available TCP or UDP port above 1025, RPC vulnerabilities extend to these ports as well. This has become exacerbated with the popularity of Windows Messenger. LURHQ (www.lurhq.com/popup_spam.html) observed that large blocks of IP addresses recently began targeting ports 1026–1029 with anonymous pop-up spam in response to ISP filtering port 135. Independent confirmation has been reported by SANS (see isc.sans.org/port_details.php?port=1026). In addition, these ports are vulnerable to buffer overflow attacks before XP SP2. If you don't need Windows Messenger, closing ports 1026–1029 is well advised.

Beyond the direct vulnerabilities, NetBios has also been used by media harvesters such as Scour (www.oldversion.com/program.php?n=scour). Though bankrupted in 2000 after being sued for copyright infringement, copies of the software remain on the Internet.

Another feature related to NetBios is the Windows Internet Naming Service (WINS) that operates on TCP/UDP port 42. WINS handles name resolution on Windows computers so that the translations from NetBios names to IP addresses are stored internally on Windows computers that share resources. The vulnerability arises from the WINS replication service whereby Windows computers share these translation tables. This service is vulnerable to packet-crafting attacks that substitute a corrupted version of the address table for the legitimate version, thereby misdirecting traffic to hacker-controlled computers (see www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1080). Blocking TCP/UDP port 42 at the firewall avoids this vulnerability.

Most users are unaware that a large number of services are running, much less *why*.

TCP/UDP port 3389, the Windows Remote Desktop Protocol port is also worrisome. This protocol provides remote display and input capabilities for Windows applications running on servers. Though the capabilities are limited to SVGA display resolution and clipboard data transfer (not to mention being somewhat slow), features like printer redirection, virtual channels, support for remote system administration and technical support, and client-server display toggling make RPC a popular choice for remote connections. We note that with the exception of the Media Center Edition of XP, RDP is disabled by default.

The problem with WTS is that RDP has been shown to be vulnerable to denial-of-service attacks by malformed packets up to and including Service Pack 2 of Windows XP. Though Microsoft announced both a security advisory and patch in July, unpatched computers remain vulnerable. Because of the nature of this vulnerability, and the fact that RDP has been beset by vulnerability reports for several years, it is recommended that port 3389 be blocked at the firewall and that RDP be disabled as well.

Unix-centric. Most Unix systems, regardless of brand, offer SMTP mail services through TCP port 25. If the system is not actually a mail relay or server, then you likely do not need this service running at all. The main exposure lies in the failure to continuously patch the system. It is a common misconception that an SMTP server must be running for a local user or service to send email out of a Unix box. This is simply not true. The recommended solution is to block TCP 25 at the firewall.

Ports 135–139, both TCP and UDP, have become extremely common on Unix machines, allowing them to interact with their Windows counterparts using the Samba service. While this is an excellent facility to allow for file and printer sharing, or even authentication services, it is also very common to find that the services permit anyone on the network to connect to the server. This can lead to the same sorts of exposures created by unprotected shares on Windows systems, and can also be used to perform brute force authentication attempts or password guessing against the service to determine user names and passwords. The exposure is further amplified by the fact that the user names and passwords are often synchronized between the Unix and Windows systems in order to allow for seamless integration. This means that once one system is compromised, all systems with that same user are also compromised. Ports 135–139 should be closed unless these services are critical.

TCP port 513 is typically used for the Rlogin service. This service, along with RSH and RCP, is considered ill-advised for use in public environments. Most security practitioners would also recommend that they be eliminated on private networks as well in favor of a service like SSHv.2, which can be used to perform all of the same "R" utility functions in a more secure manner (namely encrypted, possibly stronger authentication). However, many college and university systems have these services enabled and may be actively using these services for convenience. If these services are offered, there must be some independent network-level control in place since these services in particular make their authentication decisions based on the source address from which a connection arrives. Fooling these services is generally considered to be a trivial exercise. Adding network-level access controls at routers, firewalls, and switches to limit communication with servers via TCP 513 is recommended.

TCP port 22 is used by the Secure Shell service. It may be surprising to see that a service designed for security is listed in our set of ports to watch carefully, but the fact is this service is a great target for attackers. Not only do we have the same vulnerabilities that exist with all ports when the patch level is not maintained, but the attacker also has the potential to create an encrypted session, thereby preventing monitoring systems from performing signature alerting on the contents of the traffic. Another risk is the potential for brute forcing credentials. SSH systems are commonly configured to perform username/password authentication. This means that by determining a username for a system (possibly from an email address), the attacker can try brute force login attempts by bypassing the lockout restrictions that normally apply, since SSH is usually not integrated into the typical user authentication. One way to limit this exposure is to either limit access to port 22 or use certificate-based authentication (or both).

TCP port 389 is typically used by the LDAP service. The purpose of this service is to provide directory services to allow lookups of names, email address, public keys, and so forth. Depending on what sort of information is stored in the LDAP server and how it is secured, it may be possible for someone to browse or query the LDAP server to recover usernames to be used in brute force guessing attempts or email addresses to fill up your mailbox with spam. Blocking TCP and UDP port 389 is recommended, along with related LDAP ports 3268, 3269 and 636, especially in Windows environments.

Telnet has been around for years. It is extremely common to find this service in frequent use on internal and university networks. Since telnet is a plain-text protocol, all usernames and passwords are readable off the wire. SSHv2 is considered to be the optimal way to remotely connect to a system—even if the network is internal. There is a popular misconception that a switched network prevents someone from intercepting telnet communication with a sniffer. This is not true. Tools like Ettercap and DSniff, with ARP spoofing capabilities make this possible. Telnet ports TCP/UDP 23 should be blocked at the firewall, and internal use should be restricted.

The FTP service, which typically runs on port 21, falls into the same category as telnet and is just as ancient. While FTP might be useful for public, anonymous file repositories, it is unwise to use FTP with actual usernames and passwords. The previous discussion of plain-text protocols is relevant here as well. Consider replacing this service with SSH and using SCP or SFTP for your file transfers.

Port 161 UDP is the Simple Network Management Protocol (SNMP) service. You may or may not find this service running on your system depending on the brand and purpose, though it is very common to find it on routers, switches, and printers. The danger here is that the service can be used to poll the system for a large variety of configuration information and may even be used to reconfigure the system. To protect against this, SNMP provides for password authentication in the form of community strings. Except for the newer forms of SNMP, all of these strings are sent in plain text across the network. Additionally, SNMP servers typically have no capability to limit requests after a predetermined number of failed attempts. This means that leaving these ports open provides yet another opportunity for brute force guessing of passwords. Please note that this is much easier than username and password guessing since there is no username; thus the complexity of the problem is vastly simplified.

Port 514 UDP is usually a syslog service. This service allows for remote systems to send log messages into the local system, adding them to the local syslog. While this is a great system, there is no validation of the messages. Additionally, since the service runs over UDP, it is trivial to spoof messages to be added to the syslog service. One attacker tactic is to either inject misleading messages into your system log or to simply inject so many messages that your disk fills, preventing you from logging the actual attack that may follow. Network-level filtering of port 514 goes a long way to limiting where the damage can come from, but another solution might be to upgrade to SyslogNG. This package allows you to run syslog over TCP and even allows you to add authentication and encryption capabilities, preventing both of these attacks.

OS-independent. TCP ports 20–23, while more typically found in Unix environments, are also included in this category. Within this group, FTP (ports 20 and 21) are most commonly found on servers (whether Windows- or Unix-based), while ports 22 and 23 (SSH and telnet) are found on the majority of network-available devices. For this reason, all of the same issues discussed in the Unix section apply here, but possibly with greater risk. While the brute force or sniffing risks still exist, it is far less likely that any effort will be put into patching your network-attached printer running a telnet service. The same goes for a switch that is running either SSH or telnet (or both) deep within the network. In the case of the printer, patches for the device simply may not be available. For the switch, patches might be overlooked if these are not core devices; the routers would tend to get all of our attention.

Similarly, UDP ports 161 and 514 are vulnerable for Unix and systemwide networks. Most network-attached devices provide for some sort of SNMP management that exposes our networks and devices to, at the very least, pernicious querying of these devices for information. In the worse case, this service may be used to guess the private community string or password that can be used to reconfigure the device or query sensitive information from the device.

TCP port 79 would have fallen into our Unix section, but most modern Unix systems disable the finger service by default. The service allows remotely querying a system to discover who is currently logged in and where they are logged in from. While this service is far from ubiquitous, it still appears enough to be mentioned as a potential risk. Over time, many network-attached devices took their service cues from Unix. This means that many devices, routers as an example, implement the finger service. There are at least two dangers here. The obvious danger is that if the service is running, malicious individuals could potentially gather valid user IDs for the systems in question. Connected with this is the potential for a trust relationship between devices or at the very least a high probability that your user ID on one box is the same as your user ID on another box, solving half of the username/password guessing equation. The second less obvious issue is that unnecessary services tend to go unpatched. Should a vulnerability be discovered in the finger service for whatever device it is running on, we have an instant vulnerability. This service can sometimes be found running on routers, switches, and RIPs (Rasterizing Image Processors, often embedded or external Unix systems connected to color laser copiers and other network-attached copiers).

TCP port 515 is commonly used as the Unix-line printer daemon. For the moment, however, let's abstract this from port 515 and make this more generally network printer ports, which can be varied. Most business and academic environments are rife with network-attached printers. This proves to be the most economical way to deploy higher-end printers to improve overall productivity and maintain an acceptable level of usability. In order for these printers to work, however, they will naturally need some sort of print server running internally. As an aside, some of these printers may be configured with an external print server, whether this is a network adapter or, say, a Windows print server. Either way, there is still a network queue available somewhere for submitting jobs to the printer. Providing these kinds of services requires a significant level of computational power on the part of the printer or device managing the queue. For this reason, these devices can represent significant points of vulnerability. Why so? To abstract the problem for just a moment, let's consider the Linksys WRT-G WiFi box. This useful little wireless hub acts more or less as a wireless bridge or router, depending on how we configure it. To support this there are management interfaces available over the network (most using HTTP), and an embedded OS. One of the most interesting developments with regard to this box was the creation of a flashable open source version of Unix. In fact, at a recent hacker convention, one of the talks discussed how to turn the WRT-G into a penetration-testing platform. While we're not aware of any publicly available distributions to perform the same tasks on a printer, recent printers have a flash-based OS installed. If someone were to compromise your network switch and start collecting traffic with a sniffer we would have an exposure, but at least the attacker would have to reassemble everything into meaningful information. If the printer were to be compromised we might even have a larger problem since the data arrives already assembled and ready to print.

With the proxy misconfigured, an attacker could use the proxy itself to perform probes, in essence turning our proxy server against us.

TCP port 1080 is typically used for the SOCKS proxy service. As we did with port 515, we wish to abstract this port out to all network proxies. Now these proxies can commonly be found on ports 80, 1080, 3128, 8080 and even others, but the port number is not what's most important. The real issue is how these systems are configured.

The purpose of a proxy is typically to allow someone behind the proxy to connect to something outside of the proxy with the proxy doing all of the actual connecting, perhaps even inspecting the data that is returned. Some use proxies to attempt to anonymize their activities, concealing their actual IP address and

URL Pearls

Additional detail on TCP/IP ports is available from several sources. The definitive site is the Internet Assigned Numbers Authority (IANA) at www.iana.org. The official port assignments may be found therein at www.iana.org/assignments/port-numbers. Two derivative Web sites provide an easy-to-use interface to the IANA list: the Internet Ports Database at www.portsdb.org, and our own Internet Ports Pass at ccr.i2.nscee.edu/ports.

One of the best and most reliable sources of information on port vulnerabilities is the SANS Internet Storm Center at isc.sans.org. This site is an exceedingly rich repository of reports, charts, databases, statistics, and archives on all aspects of port vulnerabilities. The port lookup feature provides ready access to port-specific vulnerabilities. The "Top 20" list is also a useful resource (www.sans.org/top20).

For those who really want to get serious about port security, we recommend www.snort.org. The page says it all: "Snort is the de facto standard for intrusion detection/prevention."

For a refresher on packet-based networks, including a discussion of appliances and ports, the short animated film *Warriors of the Net* (www.warriorsofthe.net/movie.html) is entertaining and well worth the 12 minutes invested viewing the film.

perhaps their browser type. The danger arises with misconfigured proxies. If the proxy is not configured correctly, the proxy might be what is called an open proxy, where anyone on the Internet can connect to a proxy and use it to connect anywhere else. The worst kind of misconfiguration for a proxy occurs when it allows someone outside the proxy to connect to internal machines through the proxy. If the proxy is serving as a security device or firewall, attackers would not normally be able to directly probe the network. With the proxy misconfigured, however, an attacker could use the proxy itself to perform probes, in essence turning our proxy server against us.

Conclusion

While we've just scratched the surface of port vulnerabilities, the examples here should be convincing for even the risk-tolerant that continuous oversight of their firewalls and routers are in order. C

Hal Berghel (www.berghel.net) is associate dean of the College of Engineering and an Erskine Fellow at the University of Canterbury. He is also the director of the Center for Cybermedia Research and the director of the National Identity Theft and Financial Fraud Research and Operations Center at the University of Nevada, Las Vegas.

David Hoelzer is a director at the Internet Forensics Lab at CCR and a faculty member of the SANS Institute. He is the owner of CyberDefense (www.cyber-defense.org), a computer security and forensics consultancy.