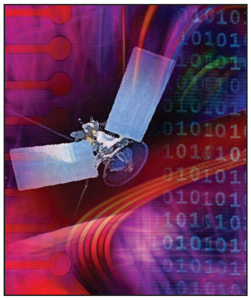


Network Forensics Analysis

Vicka Corey, Charles Peterman, Sybil Shearin, Michael S. Greenberg, and James Van Bokkelen • *Sandstorm Enterprises*

A network's physical layer is deceptively quiet. Hub lights blink in response to network traffic, but do little to convey the range of information that the network carries. Analysis of the individual traffic flows and their content is essential to a complete understanding of network usage. Many tools let you view traffic in real time, but real-time monitoring at any level requires significant human and hardware resources, and doesn't scale to networks larger than a single workgroup. It is generally more practical to archive all traffic and analyze subsets as necessary. This process is known as reconstructive traffic analysis, or *network forensics*.[1] In practice, it is often limited to data collection and packet-level inspection; however, a network forensics analysis tool (NFAT) can provide a richer view of the data collected, allowing you to inspect the traffic from further up the protocol stack.[2]

The IT industry's ever-growing concern with security is the primary motivation for network forensics. A network that has been prepared for forensic analysis is easy to monitor, and security vulnerabilities and configuration problems can be conveniently identified. It also allows the best possible analysis of security violations. Most importantly, analyzing a complete record of your network traffic with the appropriate reconstructive tools provides context for other breach-related events. For example, if your analysis detects a user account and its Pretty Good Privacy (PGP, www.pgp.com/index.php) keys being compromised, good practice requires you to review all subsequent activity by that user, or involving those keys.

In some industries, laws such as the Health Insurance Portability and Accountability Act (HIPAA, http://cms.hhs.gov/hipaa) regulate monitoring the flow of information. While it is often difficult to balance what is required by law and what is technically feasible, a forensic record of network traffic is a good first step.

Security and legal concerns are not the only reasons to want a fuller understanding of your network traffic, however. Forensics tool users have reported many other applications. If your mail server has lost several hours' or days' worth of received messages and traditional backup methods have failed, you can recover the messages from the recorded traffic. Similarly, the forensics record allows unhurried analysis of anomalies such as traffic spikes or application errors that might otherwise have remained hearsay.

NFAT's Place and Purpose

Given that firewalls and intrusion-detection systems (IDSs) are well-established tools for network security, what is NFAT's role? Will it replace these tools or complement them?

A typical IDS attempts to detect activity that violates an organization's security policy by implementing a set of rules describing preconfigured patterns of interest. These rules are both a strength and a weakness: An IDS can detect certain incidents reliably, but no rule set can detect all possible intrusions.

A typical firewall allows or disallows traffic to or from specific networks, machine addresses, and port numbers, but protocols that circumvent port-based security are increasingly common. Consider Yahoo Messenger (www.venkydude.com/articles/yahoo.htm), which will move to port 23 (well known as the Telnet[3] port) if its default port (5050) is blocked. Thus, it could circumvent a firewall's block of port 5050. An NFAT, on the other hand, would identify the connection on port 23 as Yahoo Messenger by its content.

An NFAT synergizes with IDSes and firewalls in two ways: It preserves a long-term record of network traffic, and it allows quick analysis of trouble spots identified by the other two tools. Access to an NFAT lets you decide what traffic is of interest post hoc (for example, the last two weeks' worth

1089-7801/02/$17.00 ©2002 IEEE

of e-mail sent by an employee who has disappeared and whose machine has been wiped clean) and to analyze that traffic quickly and efficiently. This analysis (via stream reassembly) reduces the severity of the *audit reduction problem*, first identified by James P. Anderson,[4] in which the volume of irrelevant information obscures the information of interest. You can use network forensics analysis capabilities to research everything connected with the incident to get "the rest of the story."

As an essential complement to existing security systems, an NFAT must perform three tasks well. It must *capture* network traffic; it must *analyze* the traffic according to the user's needs; and it must let system users *discover* useful and interesting things about the analyzed traffic.

Traffic Capture

Traffic relevance, data integrity, and packet capture rate are primary concerns when positioning an NFAT on a network.

Before placing an NFAT's traffic capture component, you must understand the network's architecture. The first consideration must be one of policy: Where is the traffic of interest? Most organizations are more concerned with traffic entering and leaving the organization than with internal use on a backbone LAN. The backbone's relatively high volume makes it more difficult to capture or archive traffic long enough to analyze events after the fact. A focus on externally accessible servers favors a monitoring point on the external (demilitarized zone, or DMZ) network. Interactions between employees and customers, partners, competitors, and so on, are often better served by a monitoring point just inside the firewall. Once you've chosen the traffic of interest to your organization, you can position the NFAT to collect the relevant packets.

Performance

For effective network forensics, an NFAT must maintain a complete record of network traffic. Some network implementations accommodate this more easily than others. For example, typical 10-Mbit Ethernet hubs send all packets down all physical segments on a given subnet. Switched networks, or networks with smart bridges, on the other hand, might or might not have a port on which all packets are visible. (Such interfaces are called *mirror ports*, and are more common on high-end switches than low-end ones.)

Also important is packet capture rate. The forensics machine is potentially interested in all the traffic that goes by. Unlike a typical network host, it cannot let its interface hardware discard packets not addressed to it. A successful NFAT must be able to capture and store traffic from a fully saturated network for later analysis.

Further complicating data integrity is the fact that an NFAT should never participate in the traffic it monitors. In a typical TCP session, if a machine at one end of a connection misses a packet, it can expect that the information will be resent. The NFAT does not have this luxury; it cannot request packet retransmission. Thus, a successful NFAT must be a very capable eavesdropper.

Performing all of these tasks well takes resources. Reliably capturing the packet from the network requires an interface that doesn't lose packets as the network approaches saturation. Writing captured traffic to disk requires a system bus and storage subsystem that can keep up with network speeds. Storage size plays the final critical role: the amount of storage dedicated to traffic capture determines how far into the past you have data for forensic analysis.

An NFAT synergizes with IDSes and firewalls in two ways: It preserves a long-term record of network traffic, and it allows quick analysis of trouble spots.

Under some circumstances, an NFAT might be able to eliminate irrelevant traffic by applying a filter as it captures traffic. This can alleviate storage or performance concerns, but at some cost: The more traffic an NFAT discards, the longer the interval in which it can archive traffic, but the smaller the scope of possible post hoc analysis. Also, filtering at the collection layer sometimes increases processing overhead to the point that packets are dropped.

Capture Software

The most popular packet-collection software is currently tcpdump (www.tcpdump.org/tcpdump_man.html),[5] which is available to the root user on most Unix platforms as well as under Windows. Tcpdump is useful for collecting traffic when you know the activity of interest is on the wire. It contains no facility for disk management, however, and will blithely fill the available disk space and might crash the machine if left unattended.

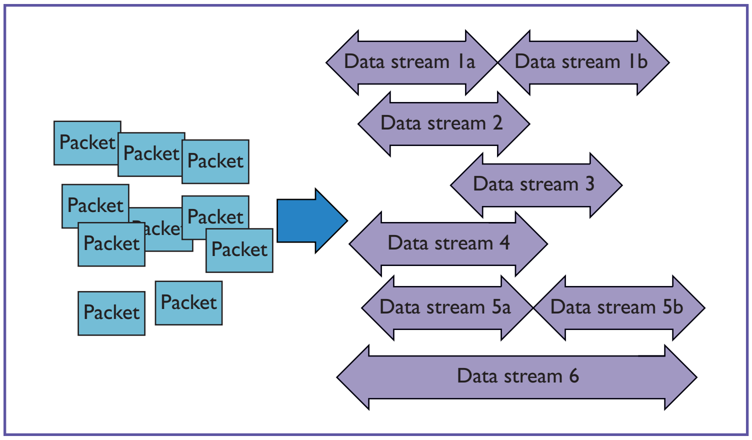


Figure 1. Sessionizing traffic. A network forensics tool organizes packets into individual transport-layer connections between machines, allowing it to analyze network traffic layer by layer.

While most older network monitors use proprietary file formats, the advent of NFAT systems has sparked a movement toward making tcpdump file format a de facto standard. This compatibility lets users export data for analysis with other tools, and also lets them import and analyze data collected using tcpdump on other systems. An NFAT's capture system can also incorporate FIFO deletion of the oldest data to maintain a reasonable amount of free space for analysis. This can improve performance, as file systems in general slow down as free space disappears.

Traffic Analysis

The process of archiving network traffic generates the first layer of forensic information: namely, traffic load over time. This is a relatively trivial matter, and several easier ways to get this information exist. But it is often a useful guide to further analysis. Perhaps there was a five-minute spike in traffic at 23:42 on Saturday. What was in that traffic spike?

Sessionizing Captured Traffic

Most network traffic is a two-way conversation between machines. In the captured network traffic, unrelated packets appear in the order they were transmitted over the wire. A network forensics tool should organize the packets into individual transport-layer connections between machines (often referred to as *sessionizing*), as Figure 1 shows.

Researchers have performed useful analyses with unsessionized data. In fact, most network-monitoring tools support selection and searching on roughly the same level as IDSes and firewalls (that is, based on IP addresses, port numbers, and scanning individual packets for data signatures of interest). Such tools easily miss protocols on nonstandard ports, however, and might find it difficult to come up with workable signatures where the data is subject to transfer encoding or encryption, or where the signatures are split between packets.

By first sessionizing the data, an NFAT can peel each connection apart, analyzing layer after layer of protocol and content. To reassemble the connections, the sessionizing process must be aware of the transport layer. During reassembly, more forensic details emerge. The *artifacts* – missing or retransmitted data, transport-layer protocol errors, and so on – become visible, providing valuable security and debugging information. Once reassembly is complete, traffic patterns between machines become visible. Which machines are talking to each other, and on which ports?

Protocol Parsing and Analysis

Researchers commonly perform protocol analysis "by hand." Run `tcpdump` to collect traffic, then `strings` to pull text from the traffic stream, and `grep` to find specific words or phrases in the recovered strings. Searching for the string `get` this way can turn up quite a bit of information about the network's Web traffic. A session that contains the string `quit` could be an FTP control session, or it might be a post office protocol (POP3)[6] session, or perhaps a network news transfer protocol (NNTP)[7] session. If it contains the string `privmsg`, it is likely to be an Internet relay chat protocol (IRC)[8] session. A TCP connection running on port 23 that contains Telnet interpret-as-command sequences (IACs) is probably Telnet, but a session running on port 23 containing the string YMSG and a lot of "c0 80" record separators is more likely to be Yahoo Messenger. Once you have identified one or more sessions of interest, you can extract them from the raw data by rerunning `tcpdump` with an appropriate filter.

A more efficient approach to uncovering this kind of information is expert-system analysis on the sessionized traffic, which lets you analyze information from each network layer. The expert system can use various well-known features of the traffic content to identify it. Port numbers can offer hints but data stream contents are definitive. Once the system has identified the data, it might need to perform an additional transform to extract a component of interest or undo a transfer encoding. Multiple layers of encapsulation or encoding can be undone in this manner.

Not only must the system evaluate the individ-

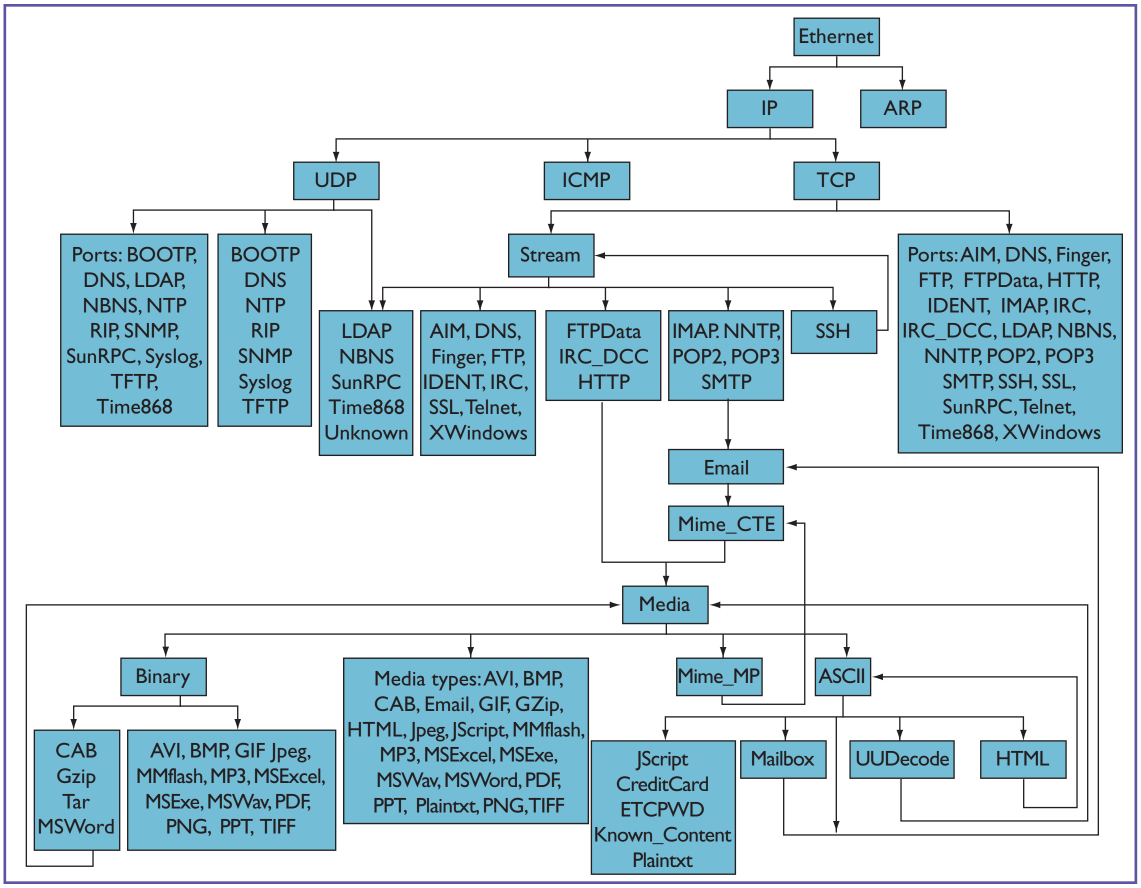

Ethernet
IP
ARP
UDP
ICMP
TCP
Ports: BOOTP, DNS, LDAP, NBNS, NTP RIP, SNMP, SunRPC, Syslog, TFTP, Time868
BOOTP DNS NTP RIP SNMP Syslog TFTP
Stream
Ports: AIM, DNS, Finger, FTP, FTPData, HTTP, IDENT, IMAP, IRC, IRC_DCC, LDAP, NBNS, NNTP, POP2, POP3 SMTP, SSH, SSL, SunRPC, Telnet, Time868, XWindows
LDAP NBNS SunRPC Time868 Unknown
AIM, DNS, Finger, FTP, IDENT, IRC, SSL, Telnet, XWindows
FTPData IRC_DCC HTTP
IMAP, NNTP, POP2, POP3 SMTP
SSH
Email
Mime_CTE
Media
Binary
Media types: AVI, BMP, CAB, Email, GIF, GZip, HTML, Jpeg, JScript, MMflash, MP3, MSExcel, MSExe, MSWav, MSWord, PDF, PPT, Plaintxt, PNG, TIFF
Mime_MP
ASCII
CAB Gzip Tar MSWord
AVI, BMP, GIF Jpeg, MMflash, MP3, MSExcel, MSExe, MSWav, PDF, PNG, PPT, TIFF
JScript CreditCard ETCPWD Known_Content Plaintxt
Mailbox
UUDecode
HTML

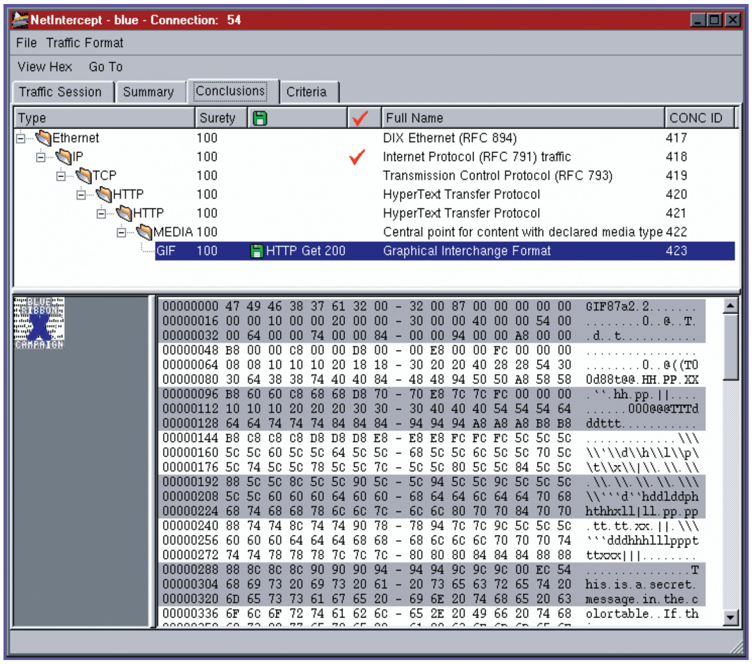
Figure 2. Modules in a parse graph. The set of modules accepts a layer-2 data stream at the Ethernet node, from which it derives higher-layer streams through layer 7.

ual connections' content, but it must also correlate the connections with each other. One HTTP connection might be transferring an image requested by an HTML file fetched over another connection. Alternatively, an HTTP `get` might be a request from a Gnutella (www.gnutelladev.com/protocol/gnutella-protocol.html) agent to download a file, and thus related to a Gnutella control session. An IRC connection might advertise a file's availability, and the only way to identify a connection that fetched the file as a direct client-to-client connection (DCC) is to correlate it with the corresponding advertisement.

Expert-system network traffic analysis gains power when we extend the concept past the International Organization for Standardization (ISO) seven-layer model and into the data stream. Our analysis engine, NetIntercept (www.sandstorm.net/netintercept), includes modules that recognize e-mail messages, uu-encoded data, Tar files, various image formats, and Gzip-compressed data streams. By arranging the modules into a directed graph that might contain loops, the system can find successive layers of content. By this means, forensic analysis lets you explore and understand data that was unintelligible at the packet-sniffer level.

Figure 2 shows the graph of a set of modules for parsing Internet traffic. This set accepts a layer-2 data stream at the Ethernet node, from which higher-layer streams are derived through layer 7. Most of the nodes below `Media` are concerned with recognizing layer 7 objects, but `CreditCard` and `KnownContent` extract object attributes.

Consider a captured FTP data session that transferred a Tar file compressed with Gzip. The module that understands Gzip identifies and uncompresses the data stream. Another module similarly identifies the resulting tar-format file and extracts its contents as individual streams. A module that recognizes Microsoft Word documents identifies three of these streams and extracts the text for further processing. Another module identifies 11 e-mail messages and extracts their content as text streams. Finally, leaf nodes in the graph scan the text streams for keywords and credit-card numbers and label the substreams in which they find them.



Type	Surety	🖫	✓	Full Name	CONC ID
Ethernet	100			DIX Ethernet (RFC 894)	417
IP	100		✓	Internet Protocol (RFC 791) traffic	418
TCP	100			Transmission Control Protocol (RFC 793)	419
HTTP	100			HyperText Transfer Protocol	420
HTTP	100			HyperText Transfer Protocol	421
MEDIA	100			Central point for content with declared media type	422
GIF	100	🖫 HTTP Get 200		Graphical Interchange Format	423

```
00000000 47 49 46 38 37 61 32 00 - 32 00 87 00 00 00 00 00  GIF87a2.2.......
00000016 00 00 10 00 00 20 00 00 - 30 00 00 40 00 00 54 00  ........0..@..T.
00000032 00 64 00 00 74 00 00 84 - 00 00 94 00 00 A8 00 00  .d..t...........
00000048 B8 00 00 C8 00 00 D8 00 - 00 E8 00 00 FC 00 00 00  ................
00000064 08 08 10 10 10 20 18 18 - 30 20 20 40 28 28 54 30  ........0..@((T0
00000080 30 64 38 38 74 40 40 84 - 48 48 94 50 50 A8 58 58  0d88t@@.HH.PP.XX
00000096 B8 60 60 C8 68 68 D8 70 - 70 E8 7C 7C FC 00 00 00  .``.hh.pp.||....
00000112 10 10 10 20 20 20 30 30 - 30 40 40 40 54 54 54 64  ......000@@@TTTd
00000128 64 64 74 74 74 84 84 84 - 94 94 94 A8 A8 A8 B8 B8  ddttt...........
00000144 B8 C8 C8 C8 D8 D8 D8 E8 - E8 E8 FC FC FC 5C 5C 5C  .............\\\
00000160 5C 5C 60 5C 5C 64 5C 5C - 68 5C 5C 6C 5C 5C 70 5C  \\`\\d\\h\\l\\p\
00000176 5C 74 5C 5C 78 5C 5C 7C - 5C 5C 80 5C 5C 84 5C 5C  \t\\x\\|\\.\\.\\
00000192 88 5C 5C 8C 5C 5C 90 5C - 5C 94 5C 5C 9C 5C 5C 5C  .\\.\\.\\.\\.\\\
00000208 5C 5C 60 60 60 64 60 60 - 68 64 64 6C 64 64 70 68  \\```d``hddlddph
00000224 68 74 68 68 78 6C 6C 7C - 6C 6C 80 70 70 84 70 70  hthhxll|ll.pp.pp
00000240 88 74 74 8C 74 74 90 78 - 78 94 7C 7C 9C 5C 5C 5C  .tt.tt.xx.||.\\\
00000256 60 60 60 64 64 64 68 68 - 68 6C 6C 6C 70 70 70 74  ```dddhhhlllpppt
00000272 74 74 78 78 78 7C 7C 7C - 80 80 80 84 84 84 88 88  ttxxx|||........
00000288 88 8C 8C 8C 90 90 90 94 - 94 94 9C 9C 9C 00 EC 54  ...............T
00000304 68 69 73 20 69 73 20 61 - 20 73 65 63 72 65 74 20  his.is.a.secret.
00000320 6D 65 73 73 61 67 65 20 - 69 6E 20 74 68 65 20 63  message.in.the.c
00000336 6F 6C 6F 72 74 61 62 6C - 65 2E 20 49 66 20 74 68  olortable..If.th
```

Figure 3. Information hidden in an image file.The conclusion tree lists conclusions that the expert system made about this particular connection.

Handling Encrypted Data

FTP is an insecure protocol that typically includes a cleartext username/password pair, which anyone collecting network traffic can easily intercept, read, and fraudulently reuse. The security-conscious often replace FTP with Secure cp (SCP), which provides similar functionality (file transfer) within a secure shell (SSH, www.ietf.org/html.charters/secsh-charter.html or www.openssh.org) session encrypted on the wire.

In a high-security environment, however, it is not necessarily appropriate to use protocols that you cannot monitor and audit. Although you might want to encrypt the complete session to keep it safe from prying eyes during transfer, you might need to allow authorized personnel to review those transfers as they were on the wire. Under some circumstances, you can achieve this by configuring one end point to log the stream's unencrypted contents. This approach is not widely used. In addition to introducing several new security vulnerabilities, it doesn't scale well to large networks.

An NFAT can support this review requirement nonintrusively and with minimal security impact. To do so, either the client or server machine must use modified SSH software and be configured with the NFAT system's public key. Using this key, the client or server encrypts the ephemeral keys for each new SSH stream and transmits them using a channel within the SSH protocol (typically used for debugging). The NFAT system captures and decrypts the ephemeral keys and uses them to monitor the SSH session content. In the file-transfer example, this would let the client and server use SCP to transfer a fully encrypted file, and also let the forensics tool capture and decrypt the connection, and subsequently analyze its contents.

Interacting with an NFAT

An NFAT user interface should facilitate traffic and content inspection. The interface should let the operator accurately specify traffic that is of interest, and avoid viewing traffic that is not. Traditional network-monitoring tools support criteria for specifying traffic of interest such as connection sizes, end point media access control (MAC) or IP addresses, and TCP or user datagram protocol (UDP) port numbers. NFAT systems can extend this by allowing selection according to user or file names, specific content types, human interpretation of transferred images, and so on. An NFAT user interface should make specifying selection criteria easy and definitive.

Exploring and Reporting Findings

Network forensics can generate enormous quantities of information. Its analysis ranges from the trivial (When did host A ask host B for a network time protocol[9] update?) to the grandiose (When are the demand peaks, and which protocols contribute to them?) to the obscure (What implementations of RFC 868 violate the published standard?). This flexibility is extremely powerful. But to be useful, the NFAT must allow easy access to any particular piece of information.

Two key factors in this are searchability and visibility. Often, the user logs in to the NFAT system with a specific set of target criteria in mind and wants to efficiently determine what traffic matches them. For this to be possible, the display of available criteria must be searchable for specific values. At other times, the user is looking for anomalies rather than a specific target. In this case, both the details and the overview of the available data need to be easily viewed in various ways. In either case, selecting what you want implies eliminating the irrelevant and the unwanted. If you need to scrutinize a particular user's e-mail, you should avoid looking at everybody else's e-mail at the same time. Besides helping preserve privacy, it will reduce your workload.

Once a user has selected a set of connections for inspection, the forensics tool should allow easy access to different levels of perspective on the data.

For instance, in the "Blue Ribbon Campaign" screenshot in Figure 3, the graphical, hexadecimal, and ASCII levels are presented simultaneously. But several other views might be useful. If this graphic appeared on a Web page, what did the rest of the page look like? Did the receiving client click on it as a link? What other traffic came from that server? How often was the file downloaded? All of this information exists in the raw data, and to the extent that these questions can't be easily answered via the NFAT's interface, the tool has fallen short of its potential.

Displaying Network Traffic Content

Much of the data traveling on modern networks is visual. Our forensic analysis of sample network traffic data has revealed that the ratio of images found to total Ethernet connections is about 2:5 – a tremendously high proportion of pictures. Other data objects might be textual (a proportion of about 1:5), or most meaningful when considered as binary or hexadecimal (such as a time protocol exchange or the transfer of an executable file). These are not, of course, strictly definable categories, and it is often useful to view the contents of network traffic at multiple levels.

Consider Figure 3. At the top of the screen is a *conclusion tree*, which shows conclusions that the expert system made about this particular connection. The blue ribbon image is a GIF file, contained inside a Web page. On the left is a view of the image as an image; on the right is a hexadecimal representation of the same data; and on the far right is a hexadecimal and ASCII representation, including the text, "This is a secret message in the colortable."

Graphical display of graphical information helps us understand a network's traffic. More elaborate renditions than the above example exist, such as the reconstruction of Web pages or traffic patterns between network hosts. But the examination of data at other levels, such as ASCII and hexadecimal, can also be very valuable.

Displaying Information about the Network

In addition to examining individual traffic elements, NFATs offer higher-level approaches to understanding network content, topology, and functioning. When identifying the traffic components, an NFAT can describe them collectively as well as individually. Network forensics can illuminate issues such as bandwidth use in terms of machines, protocols, users, or content. It can summarize findings that might be of concern, such as unauthorized services, cleartext-password protocols, or implementations that violate protocol standards.

Security Issues

Operating an NFAT raises some specific security concerns, such as handling encrypted traffic (discussed previously), avoiding detection and circumvention, and protecting the sensitive data revealed by analysis.

Avoiding Detection

In 1999, L0pht Heavy Industries (a predecessor of @Stake) introduced a program called antisniff (www.@stake.com), which attempted to find other machines running packet monitors (*sniffers*) on the wire. Although network traffic collection is by definition a passive phenomenon, peculiarities in common NT and Unix TCP stacks make the machines behave differently when sniffers are running. For example, a sniffer-detector might send packets to or from bogus IP addresses and note responses or attempted inverse lookups on the addresses via DNS. Once an attacker senses a traffic monitor, it might take additional precautions such as using encryption or stealthed attacks. Or it might turn its efforts against the eavesdropping machine.

The whole point of network forensics analysis is to reduce this data to useful information.

We have handled this by isolating the traffic collection utility being monitored to make it "silent on the wire." This is most easily accomplished through a dedicated interface, disabled for sending, for traffic capture. This way, sniffer-searching software can't find it. As an additional point of security, this design approach also insulates the NFAT system from most attacks against vulnerabilities in its own protocol stack.

Protecting Data

All of the data used in forensics analysis – the packets and their contents – are available to anyone with physical access to the same wire (except, of course, shared secret cryptographic keys.) However, a dump of a single day's worth of traffic on a modestly loaded network is a great deal of raw data, and analyzing it is a significant amount of work. The whole point of network forensics analysis is to reduce this data to useful information.

Consider the problem of protocols that use cleartext passwords, such as FTP and POP3. Password collection is one simple function of network forensics analysis (though there are also specialized password-collection tools available). Finding cleartext-password services and shutting them down can be a critical step in securing a network. But if hostile users exist on a network, providing them with the results of forensic analysis can do considerable harm. Distributing the information in its reduced form – say, printing out a username/password report on a networked printer using an unencrypted protocol or sending it via e-mail – could amount to handing sensitive information out on a plate.

The "silent on the wire" model helps address this problem as well. By isolating the machine performing the forensics from the network it monitors, you reduce the opportunities to have that information sniffed in turn. Computers performing network forensics are most secure when users can access them only from their consoles (where physical security can be counted on). A lesser level of security involves multihoming the machine, with a silent interface on the monitored network and an interactive one on a private network with access limited by policy or physical barriers.

You can also choose the level of information taken from the forensic analysis. At one level, an NFAT might generate a report of cleartext-password protocols that shows server names, protocols, usernames, and passwords. Or you might prefer to have the NFAT merely report the servers and the protocols, which would suffice for identifying and closing down vulnerable services.

Conclusion

Network forensics analysis requires dedicated tools with a constant passive function. The ability to sift through gigabytes of captured network traffic and construct multiple views of that traffic benefits network security, policy enforcement, and network maintenance personnel. The NFAT's powerful traffic analysis capabilities and its search and visualization tools significantly reduce the amount of time spent identifying and fixing network problems. It is likely that any network forensics tool will uncover detailed information about your network traffic – information that is often private. Safeguarding that information must be a priority when installing an NFAT.

References

1. S. Garfinkel, "Network Forensics: Tapping the Internet," *O'Reilly Network*, www.oreillynet.com/pub/a/network/2002/04/26/nettap.html, Apr. 2002.
2. J. Postel and J. Reynolds, "Telnet Protocol Specification," Internet Engineering Task Force RFC 854, May 1983, available at http://www.ietf.org/rfc/rfc0854.txt.
3. N. King and E. Weiss, "Network Forensics Analysis Tools (NFATs) Reveal Insecurities, Turn Sysadmins into System Detectives," *Information Security*, Feb. 2002, available at www.infosecuritymag.com/2002/feb/cover.shtml.
4. J. Anderson, *Computer Security Technology Planning Study 2,* tech. report ESD-TR-73-51, Electronic Systems Division, Air Force Systems Command, Hanscom Field, Bedford, Mass., Oct. 1972.
5. Sandstorm Enterprises, "NetIntercept 1.1 White Paper," www.netintercept.com/whitepaper/, June 2002.
6. J. Myers and M. Rose, "Post Office Protocol – Version 3," IETF RFC 1939, May 1996, available at www.ietf.org/rfc/rfc1939.txt.
7. S. Barber, "Common NNTP Extensions," IETF RFC 2980, Oct. 2000, available at www.ietf.org/rfc/rfc2980.txt.
8. J. Oikarinen and D. Reed, "Internet Relay Chat Protocol," IETF RFC 1459, May 1993, available at www.ietf.org/rfc/rfc1459.txt.
9. J. Postel and K. Harrenstien, "Time Protocol," IETF RFC 868, May 1983, available at www.ietf.org/rfc/rfc0868.txt.

Vicka Corey is an engineer at Sandstorm Enterprises and an assistant in neuroscience at Massachusetts General Hospital. She received a doctorate in cognitive neuroscience from the University of Washington. Her primary research interest is the construction of models for cognitive systems.

Charles Peterman is an engineer at Sandstorm Enterprises. He received a master's degree in computer science from Tufts University, where his work focused on pattern recognition. He has worked on public key infrastructure implementations, system security, and visual programming environments.

Sybil Shearin is the NetIntercept project lead at Sandstorm. She holds a master's degree in library and information science from the University of Texas at Austin. Shearin has researched software agents, intelligent interfaces, and personalization/profiling at the MIT Media Lab.

Michael S. Greenberg is an engineer at Sandstorm Enterprises, which he helped to found. He holds a BA in computer science from Hampshire College. Greenberg has published and spoken on security, mobile agents, and cooperative multiagent systems, and reviewed for the IEEE. He is a member of the IEEE and the ACM.

James Van Bokkelen is the president of Sandstorm Enterprises. He attended the Massachusetts Institute of Technology. His research interests are interoperability and heuristic recognition of network protocols and TCP tuning. Van Bokkelen has authored and collaborated on Internet RFCs, the Packet Driver specification, and the WinSock API specification.

Readers can contact Van Bokkelen at james@sandstorm.net.