1

NETWORK SECURITY MONITORING RATIONALE

This chapter introduces the principles of *network security monitoring (NSM)*, which is the collection, analysis, and escalation of indications and warnings to detect and respond to intrusions. NSM is a way to find intruders on your network and do something about them before they damage your enterprise.

NSM began as an informal discipline with Todd Heberlein's development of the Network Security Monitor in 1988. The Network Security Monitor was the first intrusion detection system to use network traffic as its main source of data for generating alerts, and the Air Force Computer Emergency Response Team (AFCERT) was one of the first organizations to informally follow NSM principles.

In 1993, the AFCERT worked with Heberlein to deploy a version of the Network Security Monitor as the Automated Security Incident Measurement (ASIM) system. I joined the AFCERT in 1998, where, together with incident handler Bamm Visscher, I codified the definition of NSM

for a SearchSecurity webcast in late 2002. I first published the definition in book form as a case study in *Hacking Exposed, Fourth Edition*.[1] My goal since then has been to advocate NSM as a strategic and tactical operation to stop intruders before they make your organization the headline in tomorrow's newspaper.

The point of this book is to provide readers with the skills, tools, and processes to at least begin the journey of discovering adversaries. We need to recognize that incident response, broadly defined, should be a *continuous business process*, not an ad hoc, intermittent, information technology (IT)–centric activity. While NSM is not the only, or perhaps even the most comprehensive, answer to the problem of detecting, responding to, and containing intruders, it is one of the best ways to mature from zero defenses to some defensive capability. Creating an initial operational capability builds momentum for an organization's intrusion responders, demonstrating that a company *can* find intruders and *can* do something to frustrate their mission.

An Introduction to NSM

To counter digital threats, security-conscious organizations build computer incident response teams (CIRTs). These units may consist of a single individual, a small group, or dozens of security professionals. If no one in your organization is responsible for handling computer intrusions, there's a good chance you'll suffer a breach in the near future. Investing in at least one security professional is well worth the salary you will pay, regardless of the size of your organization.

This book assumes that your organization has a CIRT of at least one person, sufficiently motivated and supplied with resources to do *something* about intruders in your enterprise. If you're the only person responsible for security in your organization, congratulations! You are officially the CIRT. Thankfully, it's not costly or time-consuming to start making life difficult for intruders, and NSM is a powerful way to begin.

When CIRTs conduct operations using NSM principles, they benefit from the following capabilities:

- CIRTs collect a rich amount of network-derived data, likely exceeding the sorts of data collected by traditional security systems.
- CIRTs analyze this data to find compromised assets (such as laptops, personal computers, servers, and so on), and then relay that knowledge to asset owners.
- CIRTs and the owners of the computing equipment collaborate to contain and frustrate the adversary.
- CIRTs and computer owners use NSM data for damage assessment, assessing the cost and cause of an incident.

1. Stuart McClure, Joel Scambray, and George Kurtz, *Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition* (McGraw-Hill Osborne Media, 2003).

Consider the role of NSM in an enterprise security process. For example, Figure 1-1 shows how different security capabilities relate to one another, but not necessarily how they compare against an intruder's process.

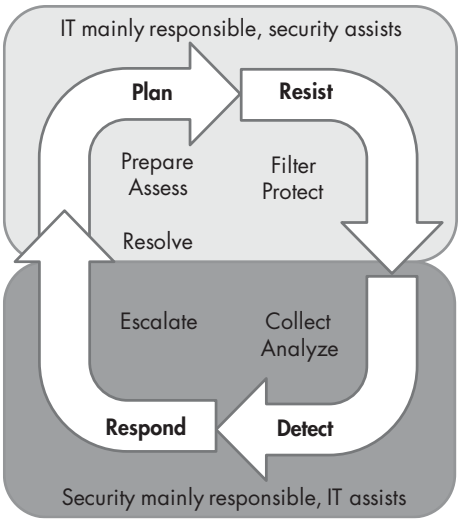

Figure 1-1: Enterprise security cycle

Does NSM Prevent Intrusions?

NSM does not involve preventing intrusions because *prevention eventually fails*. One version of this philosophy is that *security breaches are inevitable*. In fact, any networked organization is likely to suffer either sporadic or constant compromise. (Your own experience may well confirm this hard-won wisdom.)

But if NSM doesn't stop adversaries, what's the point? Here's the underappreciated good news: Change the way you look at intrusions, and defenders can ultimately frustrate intruders. In other words, determined adversaries will inevitably breach your defenses, but they may not achieve their objective.

Time is the key factor in this strategy[2] because intruders rarely execute their entire mission in the course of a few minutes, or even hours. In fact, the most sophisticated intruders seek to gain *persistence* in target networks—that is, hang around for months or years at a time. Even less advanced adversaries take minutes, hours, or even days to achieve their goals. The point is that this window of time, from initial unauthorized access to ultimate mission accomplishment, gives defenders an opportunity to detect, respond to, and contain intruders before they can finish the job they came to do.

After all, if adversaries gain unauthorized access to an organization's computers, but can't get the data they need before defenders remove them, then what did they really achieve?

I hope that you're excited by the thought that, yes, adversaries can compromise systems, but CIRTs can "win" if they detect, respond to, and contain intruders before they accomplish their mission. But if you can detect it, why can't you prevent it?

The simple answer is that the systems and processes designed to protect us aren't perfect. Prevention mechanisms can block some malicious activity, but it's increasingly difficult for organizations to defend themselves as adversaries adopt more sophisticated tactics. A team can frustrate or resist intrusions, but time and knowledge frequently become the limiting factors.

2. Security pioneer Winn Schwartau published *Time-Based Security* in 1999. I endorsed the centrality of time as presented in his book in 2005, in my post "Where in the World Is Winn Schwartau?" (*http://taosecurity.blogspot.com/2005/04/where-in-world-is-winn-schwartau-if.html*).

THE IMPORTANCE OF TIME: CASE STUDY

One real-world example shows the importance of time when defending against an intruder. In November 2012, the governor of South Carolina published the public version of a Mandiant incident response report.* Mandiant is a security company that specializes in services and software for incident detection and response. The governor hired Mandiant to assist her state with this case. Earlier that year, an attacker compromised a database operated by the state's Department of Revenue (DoR). The report provided details on the incident, but the following abbreviated timeline helps emphasize the importance of time. This case is based exclusively upon the details in the public Mandiant report.

August 13, 2012 An intruder sends a malicious (phishing) email message to multiple DoR employees. At least one employee clicks a link in the message, unwittingly executing malware and becoming compromised in the process. Available evidence indicates that the malware stole the user's username and password.

August 27, 2012 The attacker logs in to a Citrix remote access service using stolen DoR user credentials. The attacker uses the Citrix portal to log in to the user's workstation, and then leverages the user's access rights to access other DoR systems and databases.

August 29–September 11, 2012 The attacker interacts with a variety of DoR systems, including domain controllers, web servers, and user systems. He obtains passwords for all Windows user accounts and installs malicious software on many systems. Crucially, he manages to access a server housing DoR payment maintenance information.

Notice that four weeks elapsed since the initial compromise via a phishing email message on August 13, 2012. The intruder has accessed multiple systems, installed malicious software, and conducted reconnaissance for other targets, but thus far has not stolen any data. The timeline continues:

September 12, 2012 The attacker copies database backup files to a staging directory.

September 13 and 14, 2012 The attacker compresses the database backup files into 14 (of the 15 total) encrypted 7-Zip archives. The attacker then moves the 7-Zip archives from the database server to another server and sends the data to a system on the Internet. Finally, the attacker deletes the backup files and 7-Zip archives. (Mandiant did not report the amount of time needed by the intruder to copy the files from the staging server to the Internet.)

---

* South Carolina Department of Revenue and Mandiant, Public Incident Response Report (November 20, 2012) (*http://governor.sc.gov/Documents/MANDIANT%20Public%20IR%20Report%20-%20Department%20of%20Revenue%20-%2011%2020%202012.pdf*).

From September 12 through 14, the intruder accomplishes his mission. After spending one day preparing to steal data, the intruder spends the next two days removing it.

September 15, 2012 The attacker interacts with 10 systems using a compromised account and performs reconnaissance.

September 16–October 16, 2012 There is no evidence of attacker activity, but on October 10, 2012, a law-enforcement agency contacts the DoR with evidence that the personally identifiable information (PII) of three individuals has been stolen. The DoR reviews the data and determines that it would have been stored within its databases. On October 12, 2012, the DoR contracts with Mandiant for assistance with incident response.

About four weeks pass after the intruder steals data, and then the state learns of the intrusion from a third party and engages a professional incident response team. This is not the end of the story, however.

October 17, 2012 The attacker checks connectivity to a server using the backdoor installed on September 1, 2012. There is no evidence of additional activity.

October 19 and 20, 2012 The DoR attempts to remedy the attack based on recommendations from Mandiant. The goal of remediation is to remove the attacker's access and to detect any new evidence of compromise.

October 21–November 20, 2012 There is no evidence of malicious activity following remediation. The DoR publishes the Mandiant report on this incident.

Mandiant consultants, state personnel, and law enforcement were finally able to contain the intruder. Figure 1-2 summarizes the incident.

The main takeaway from this case study is that the initial intrusion is not the end of the security process; it's just the beginning. If at any time during the first four weeks of this attack the DoR had been able to contain the attacker, he would have failed. Despite losing control of multiple systems, the DoR would have prevented the theft of personal information, saving the state at least $12 million in the process.**

It's easy to dismiss a single incident as one data point, but recent statistics corroborate key elements of the case study.*** For one, the median time from the start of an intrusion to incident response is more than 240 days; that is, in most cases, victims stay compromised for a long time before anyone notices. Only one-third of organizations who contacted Mandiant for help identified the intrusions themselves.

(continued)

** The State of South Carolina reportedly owes Experian at least $12 million to pay for credit-monitoring services for breach victims. "How Will SC Pay for Security Breach?" December 3, 2012 (*http://www.wspa.com/story/21512285/how-will-sc-pay-for-security-breach*).

*** M-Trends 2013 (*https://www.mandiant.com/resources/m-trends/*).

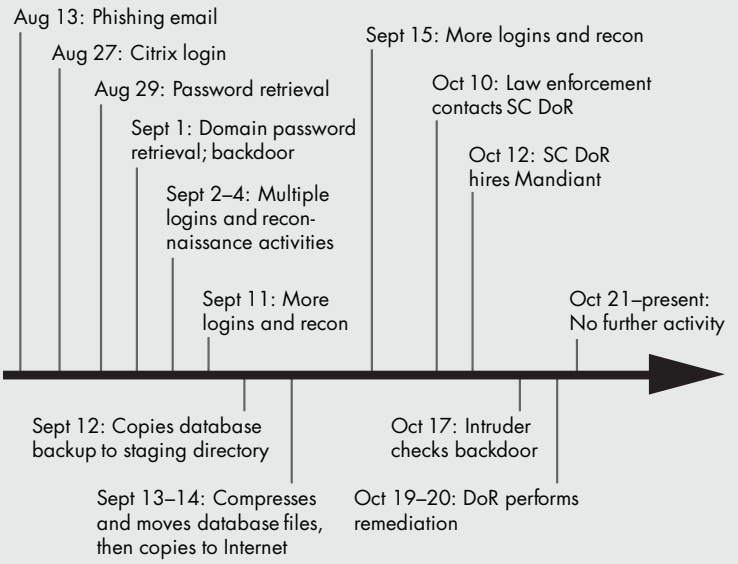

Figure 1-2: Edited timeline of South Carolina Department of Revenue incident

What Is the Difference Between NSM and Continuous Monitoring?

Continuous monitoring (CM) is a hot topic in US federal government circles. Frequently, security professionals confuse CM with NSM. They assume that if their organization practices CM, NSM is unnecessary.

Unfortunately, CM has almost nothing to do with NSM, or even with trying to detect and respond to intrusions. NSM is *threat-centric*, meaning adversaries are the focus of the NSM operation. CM is *vulnerability-centric*, focusing on configuration and software weaknesses.

The Department of Homeland Security (DHS) and the National Institute of Standards and Technology (NIST) are two agencies responsible for promoting CM across the federal government. They are excited by CM and see it as an improvement over certification and accreditation (C&A) activities, which involved auditing system configurations every three years or so. For CM advocates, "continuous" means checking system configurations more often, usually at least monthly, which is a vast improvement over previous approaches. The "monitoring" part means determining whether systems are compliant with controls—that is, determining how much a system deviates from the standard.

While these are laudable goals, CM should be seen as a complement to NSM, not a substitute for or a variant of NSM. CM can help you to provide better digital defense, but it is by no means sufficient.

Consider the differences in the ways that CM and NSM are implemented:

- A CM operation strives to find an organization's computers, identify vulnerabilities, and patch those holes, if possible.
- An NSM operation is designed to detect adversaries, respond to their activities, and contain them before they can accomplish their mission.

NOTE *For more on CM, visit NIST's website* (http://www.nist.gov/). *You will find helpful material, such as the article "NIST Publishes Draft Implementation Guidance for Continuously Monitoring an Organization's IT System Security," January 24, 2012* (http://www.nist.gov/itl/csd/monitoring-012412.cfm). *I have also posted several times on this topic at the TaoSecurity blog* (http://taosecurity.blogspot.com/)*; for example, see "Control 'Monitoring' is Not Threat Monitoring," November 23, 2009* (http://taosecurity.blogspot.com/2009/11/control-monitoring-is-not-threat.html).

How Does NSM Compare with Other Approaches?

If you're reading this book, I doubt that you operate a network without applying any security measures at all. You may wonder how your firewall, intrusion prevention system (IPS), antivirus (AV) software, whitelisting, data leakage/loss protection/prevention (DLP) system, and/or digital rights management (DRM) system work to try to stop intruders. How does this sea of security acronyms save you from attackers?

Each of these platforms is a blocking, filtering, or denying mechanism. Their job is, to the extent possible, recognize malicious activity and stop it from happening, albeit at different stages in the life cycle of an intrusion. Figure 1-3 shows how each approach might cooperate in the case of an intruder attempting to access and then steal sensitive information from an enterprise system.

These tools have various success rates against different sorts of attackers. Each generally has some role to play in the enterprise, although many organizations deploy a subset of these technologies. Their shared goal is to *control* what happens in the enterprise. When configured properly, they can operate without the need for human interaction. They just work.

Unlike these tools, NSM is not a blocking, filtering, or denying technology. It is a strategy backed by tactics that focus on *visibility*, not control. Users expect safety on the network, and they expect their security team to be aware when security controls fail. Unfortunately, failing security tools do not usually report their own weaknesses or flaws. NSM is one way to make the failure of security controls more visible.

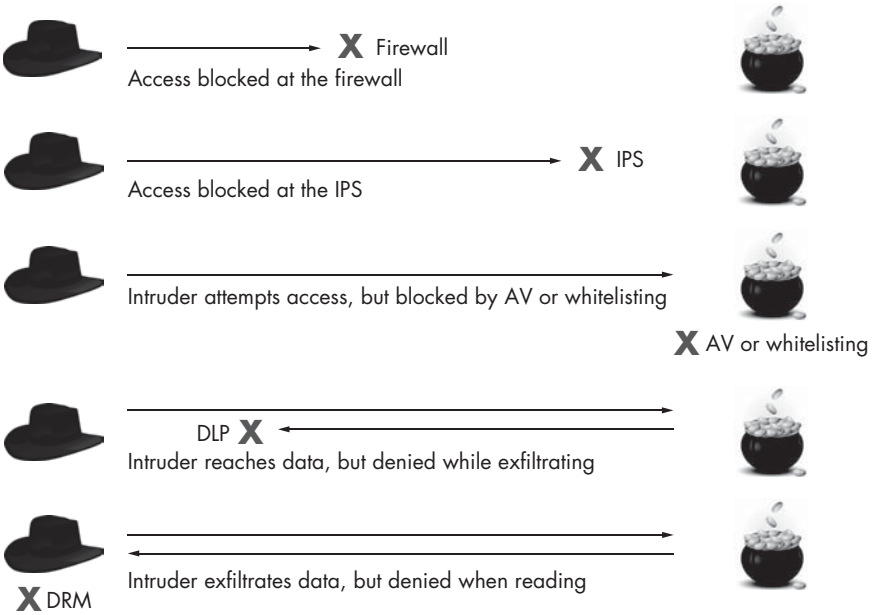

Figure 1-3: Blocking, filtering, and denying mechanisms

Why Does NSM Work?

As a system—meaning a strategy- and tactics-based operation—NSM gives us the ability to detect, respond to, and contain intruders. Yet, intruders can evade control measures that block, filter, and deny malicious activity. What makes NSM so special?

To understand this paradox, start from the perspective of the defender. Network operators must achieve perfect defense in order to keep out intruders. If an intruder finds and exploits a vulnerability in a system, the enterprise has an incident on its hands. When one sheepdog, guarding hundreds of sheep, faces a pack of wolves, at least some of the sheep will not live to see another day. The adversary "wins."

Now look at things from the intruder's perspective. Assume the adversary is not a hit-and-run offender looking for a quick strike against a weak Internet-accessible database. Rather, he wants to compromise a network, establish persistence mechanisms, and remain in the system, undetected and free to gather information at will. He is like a wolf hiding in a flock of sheep, hoping the sheepdog fails to find him, day after day, week after week, and so on.

An organization that makes visibility a priority, manned by personnel able to take advantage of that visibility, can be extremely hostile to persistent adversaries. When faced with the right kind of data, tools, and skills, an adversary eventually loses. As long as the CIRT can disrupt the intruder before he accomplishes his mission, the enterprise wins.

How NSM Is Set Up

NSM starts with the network, and if you run a network, you can use NSM to defend it. While some variations of NSM involve installing software agents on computers, this book focuses on collecting and interpreting network traffic. To implement these activities, you need to understand your network architecture and make decisions about where you most need visibility.

Consider a simple NSM deployment case. With the help of a network support team, the CIRT decides to implement an NSM operation to defend an organization's Internet-connected offices. The CIRT and the network team collaborate to select a suitable location to achieve network visibility. The CIRT asks an engineer to configure a specific network switch to export copies of traffic passing through that switch (see Figure 1-4). (In the figure, *DMZ* refers to a network conceptually "between" the Internet and internal networks, a "demilitarized zone" where outside access to systems is permitted but tightly controlled.) The CIRT then deploys a dedicated server as an NSM platform, runs a cable from the network switch to the new NSM server, and configures software to analyze the network traffic exported by the switch. Chapter 2 explains how to choose monitoring locations, so stay tuned if you're wondering how to apply this concept to your organization.

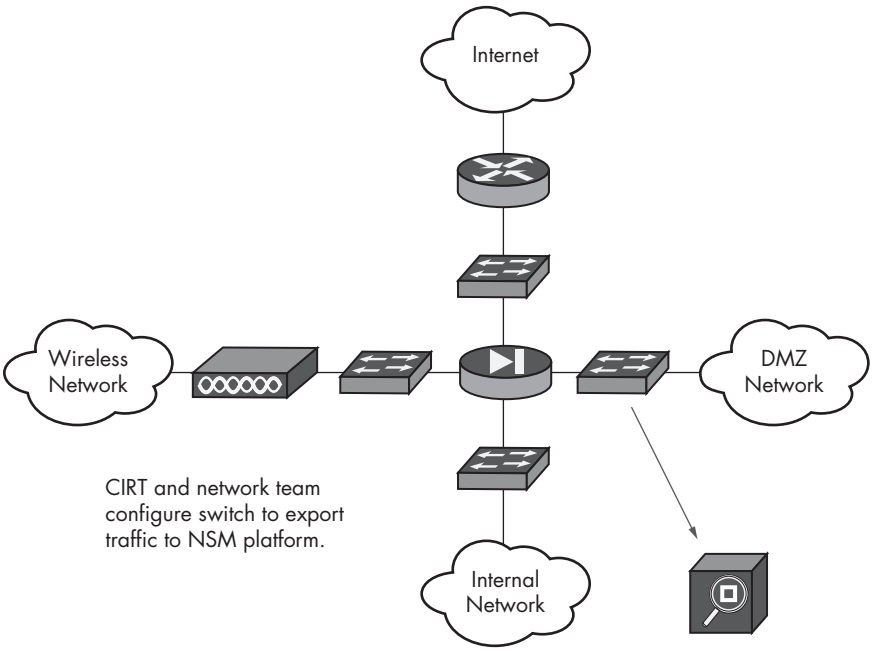

Figure 1-4: Simple network diagram and NSM platform

Installing a Tap

A better way for network and security professionals to expand visibility is to install dedicated hardware for accessing network traffic, called a *tap*. For example, Figure 1-5 shows several Net Optics taps in my lab. The top three devices are network taps, but only the hardware at top left is passing traffic. The other two taps are inactive. The devices below the taps are Cisco switches.

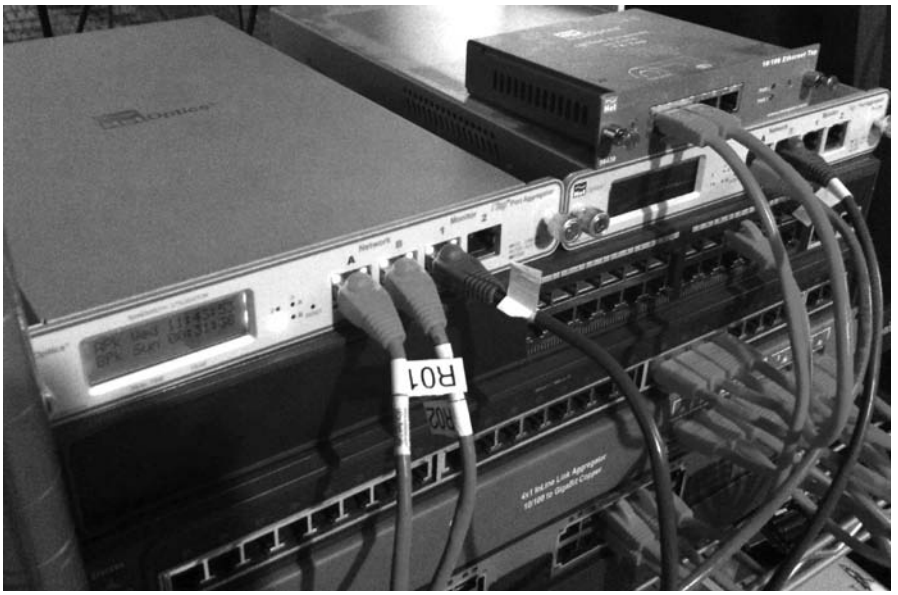

Figure 1-5: Network taps and switches

Net Optics (*http://www.netoptics.com/*) and other companies offer a wide variety of taps and related products to meet the needs of many types of organizations.

When NSM Won't Work

Regardless of how much hardware you throw at a network, if you can't observe the traffic that you care about, NSM will not work well. For example, most organizations do not conduct NSM on enterprise wireless traffic (such as 802.11 wireless local area networks, or WLANs) because the traffic from wireless node to wireless node should be encrypted, rendering NSM less effective.

This means that laptops, tablets, and other devices connected via Wi-Fi are not subject to NSM when they talk directly to each other. CIRTs *will* observe network traffic leaving the wireless segment for a wired segment. For example, when a tablet user visits a web page using a Wi-Fi connection, the NSM operation will see the activity. Node-to-node activity, though, is largely unobserved at the network level.

Similarly, CIRTs generally do not conduct NSM on cellular traffic because observing cell phone activity is outside the technical and legal mandate for most organizations. As with wireless systems, however, CIRTs will observe smartphones and cellular-capable tablets when they associate with a WLAN.

In cloud or hosted environments, NSM faces unique challenges because the service provider owns the infrastructure. While the service provider may deploy software and hardware for NSM, it usually keeps the collected data to itself. The situation is the same with ISPs and telecommunications providers.

Is NSM Legal?

There is no easy answer to the question of NSM's legality, and you should check with a lawyer. *No matter what, do not begin any NSM operation without obtaining qualified legal advice.*

In the United States, network and security teams are subject to federal and state law, such as the so-called "Wiretap Act," *U.S. Code 18 § 2511*. This includes one key provision that indicates permission for network monitoring which appears in 2511 (2)(a)(i):

> It shall not be unlawful under this chapter for an operator of a switchboard, or an officer, employee, or agent of a provider of wire or electronic communication service, whose facilities are used in the transmission of a wire or electronic communication, to intercept, disclose, or use that communication in the normal course of his employment while engaged in any activity which is a necessary incident to the rendition of his service or to the protection of the rights or property of the provider of that service, except that a provider of wire communication service to the public shall not utilize service observing or random monitoring except for mechanical or service quality control checks.[3]

Other exceptions that seem to permit monitoring involve being a party to the conversation, or obtaining consent. They appear in 2511 (2)(d):

> It shall not be unlawful under this chapter for a person not acting under color of law to intercept a wire, oral, or electronic communication where such person is a party to the communication or where one of the parties to the communication has given prior consent to such interception unless such communication is intercepted for the purpose of committing any criminal or tortious act in violation of the Constitution or laws of the United States or of any State.[4]

3. *18 USC § 2511* - Interception and disclosure of wire, oral, or electronic communications prohibited, 2511 (2)(a)(i) (*http://www.law.cornell.edu/uscode/text/18/2511#2_a_i/*).

4. *18 USC § 2511* - Interception and disclosure of wire, oral, or electronic communications prohibited, 2511 (2)(d) (*http://www.law.cornell.edu/uscode/text/18/2511#2_d/*).

The "party" and "consent" exceptions are more difficult to justify than one might expect, but they are stronger than the "necessary incident" exception.

As an example of state statutes, consider the Code of Virginia. Title 19.2, *Criminal Procedure*, contains Chapter 6, *Interception of Wire, Electronic or Oral Communications*. Section 19.2-62 in this chapter uses language that is very similar to the federal statute, which seems to allow monitoring:

> It shall not be a criminal offense under this chapter for any person . . . (f) Who is a provider of electronic communication service to record the fact that a wire or electronic communication was initiated or completed in order to protect such provider, another provider furnishing service toward the completion of the wire or electronic communication, or a user of that service, from fraudulent, unlawful or abusive use of such service.[5]

NOTE *If these laws seem onerous, the situation in the European Union (EU) tends to be "worse" from an NSM perspective. While it is important and proper to protect the rights of network users, laws in the EU seem to place a high burden on security teams. In my experience, CIRTs can deploy NSM operations in the EU, but lengthy and complicated discussions with works councils and privacy teams are required. Add a 6- to 12-month delay to any rollout plans in privacy-heightened areas.*

How Can You Protect User Privacy During NSM Operations?

Given the need to protect user privacy, it is important to manage NSM operations so that they focus on the adversary and not on authorized user activity. For this reason, you should separate the work of CIRTs from forensic professionals:

- CIRTs should perform analysis, watch malicious activity, and protect authorized users and the organization.
- Forensic professionals should perform investigations, watch fraud, and monitor abuse by authorized users, to protect the organization.

In other words, CIRTs should focus on external threats, and forensic teams should focus on internal ones. Certainly, the work of one may overlap with the other, but the key to maintaining separation is noticing when one team's work strays into the realm of the other team. Once the two have been clearly separated, users will be more likely to trust that the CIRT has their best interests at heart. (Chapter 9 expands on the operational concerns of NSM as they relate to privacy and user rights.)

5. Title 19.2, *Code of Virginia § 19.2-62*(*http://leg1.state.va.us/cgi-bin/legp504.exe?000+cod+19.2-62*).

A Sample NSM Test

Now that you know what NSM is, let's take a look at an example of activity that creates a network footprint, and then introduce how a few NSM tools see that event. Chapters 6, 7, and 8 provide details about these tools and data. The goal here is to give you a general sense of what NSM data looks like. I want you to understand how NSM and its datatypes are different from other security approaches and resources, such as firewalls, antivirus software, and application logging. The rest of the book will explain how to collect, analyze, and act on NSM data, so for now seek only to gain initial familiarity with the NSM approach.

In this example, we use the Firefox web browser to visit *http://www.testmyids.com/*, which IT professionals use to test some types of security equipment. As you can see in Figure 1-6, the page returns what looks like the output of a Unix user ID (`id`) command run by an account with user ID (UID) 0, such as a root user. This is not a real `id` command, but just a webmaster's simulation. Many tools aren't configured to tell the difference between a real security issue and a test, so visiting this website is a convenient way to catch their attention.

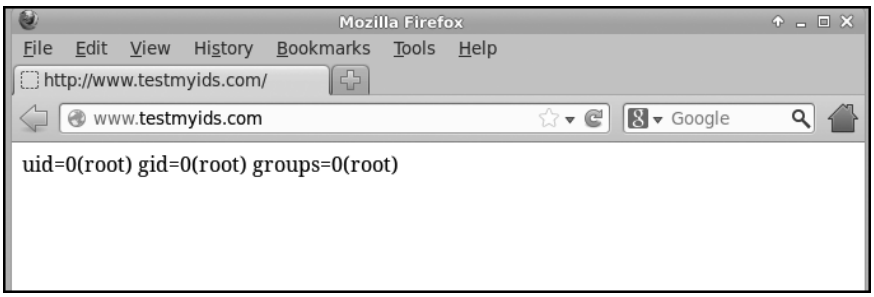

Figure 1-6: Visiting http://www.testmyids.com/ *with Firefox*

The main local evidence of a visit to the *http://www.testmyids.com/* website would probably be the user's web browser history. But on the network, the Firefox web browser and the *http://www.testmyids.com/* web server together generate three sets of data relevant to the NSM approach:

1. The browser generates a Domain Name System (DNS) request for *http://www.testmyids.com/*, and receives a reply from a DNS server.
2. The browser requests the web page, and the web server replies.
3. Finally, the web browser requests a Favorite icon from the web server, and the web server replies.

NOTE *Other traffic, such as lower-level Address Resolution Protocol (ARP) requests and replies may also occur, but they are not germane to this discussion.*

The exact mechanics of this activity are not important for this example. What is important is recognizing that all activity on a network creates traffic. NSM operators can capture this network traffic using any number of tools, and then can examine the captured data.

The Range of NSM Data

This section introduces multiple ways to analyze and view NSM data. Later chapters discuss the tools used to collect and analyze this data. NSM data may include the following:

- Full content
- Extracted content
- Session data
- Transaction data
- Statistical data
- Metadata
- Alert data

Full Content Data

For our purposes, when we collect *full content data*, we're collecting all information that passes across a network. We aren't filtering the data to collect only information associated with security alerts. We're not saving application logs. We're making exact copies of the traffic as seen on the wire.

When security analysts work with full content data, they generally review it in two stages. They begin by looking at a summary of that data, represented by "headers" on the traffic. Then they inspect some individual packets.

Reviewing a Data Summary

Listing 1-1 shows an example of data collected by running the tool Tcpdump while the Firefox web browser visited *http://www.testmyids.com/*. The IP address of the computer running the web browser is 192.168.238.152, and the IP address of the web server hosting *http://www.testmyids.com/* is 217.160.51.31. The IP address of the DNS server is 192.168.238.2.

```
19:09:47.398547 IP 192.168.238.152.52518 > 192.168.238.2.53:
 3708+ A? www.testmyids.com. (35)

19:09:47.469306 IP 192.168.238.2.53 > 192.168.238.152.52518:
 3708 1/0/0 A 217.160.51.31 (51)
```

```
19:09:47.469646 IP 192.168.238.152.41482 > 217.160.51.31.80:
 Flags [S], seq 953674548, win 42340, options [mss 1460,sackOK,TS val 75892
 ecr 0,nop,wscale 11], length 0

19:09:47.594058 IP 217.160.51.31.80 > 192.168.238.152.41482:
 Flags [S.], seq 272838780, ack 953674549, win 64240, options [mss 1460],
 length 0

19:09:47.594181 IP 192.168.238.152.41482 > 217.160.51.31.80:
 Flags [.], ack 1, win 42340, length 0

19:09:47.594427 IP 192.168.238.152.41482 > 217.160.51.31.80:
 Flags [P.], seq 1:296, ack 1, win 42340, length 295

19:09:47.594932 IP 217.160.51.31.80 > 192.168.238.152.41482:
 Flags [.], ack 296, win 64240, length 0

19:09:47.714886 IP 217.160.51.31.80 > 192.168.238.152.41482:
 Flags [P.], seq 1:316, ack 296, win 64240, length 315

19:09:47.715003 IP 192.168.238.152.41482 > 217.160.51.31.80:
 Flags [.], ack 316, win 42025, length 0

-- snip --

19:09:50.018064 IP 217.160.51.31.80 > 192.168.238.152.41482:
 Flags [FP.], seq 1958, ack 878, win 64240, length 0

19:09:50.018299 IP 192.168.238.152.41482 > 217.160.51.31.80:
 Flags [F.], seq 878, ack 1959, win 42025, length 0

19:09:50.018448 IP 217.160.51.31.80 > 192.168.238.152.41482:
 Flags [.], ack 879, win 64239, length 0
```

Listing 1-1: Tcpdump output showing headers

The output in Listing 1-1 shows only packet headers, not the content of the packets themselves.

Inspecting Packets

After looking at a summary of the full content data, security analysts select one or more packets for deeper inspection. Listing 1-2 shows the same headers as seen in the sixth packet shown in Listing 1-1 (timestamp `19:09:47.594427`), but with the layer 2 headers listed first. Layer 2 headers are just another aspect of the packet we can see. They involve the hardware-level addresses, or Media Access Control (MAC) addresses used by computers to exchange data. Furthermore, the headers are now followed by payloads, with a hexadecimal representation on the left and an ASCII representation on the right.

```
19:09:47.594427 00:0c:29:fc:b0:3b > 00:50:56:fe:08:d6, ethertype IPv4 (0x0800), length 349:
192.168.238.152.41482 > 217.160.51.31.80: Flags [P.], seq 1:296, ack 1, win 42340, length 295
        0x0000:  0050 56fe 08d6 000c 29fc b03b 0800 4500  .PV.....)..;..E.
        0x0010:  014f c342 4000 4006 ba65 c0a8 ee98 d9a0  .O.B@.@..e......
        0x0020:  331f a20a 0050 38d7 eb35 1043 307d 5018  3....P8..5.C0}P.
        0x0030:  a564 180c 0000 4745 5420 2f20 4854 5450  .d....GET./.HTTP
        0x0040:  2f31 2e31 0d0a 486f 7374 3a20 7777 772e  /1.1..Host:.www.
        0x0050:  7465 7374 6d79 6964 732e 636f 6d0d 0a55  testmyids.com..U
        0x0060:  7365 722d 4167 656e 743a 204d 6f7a 696c  ser-Agent:.Mozil
        0x0070:  6c61 2f35 2e30 2028 5831 313b 2055 6275  la/5.0.(X11;.Ubu
        0x0080:  6e74 753b 204c 696e 7578 2078 3836 5f36  ntu;.Linux.x86_6
        0x0090:  343b 2072 763a 3138 2e30 2920 4765 636b  4;.rv:18.0).Geck
        0x00a0:  6f2f 3230 3130 3031 3031 2046 6972 6566  o/20100101.Firef
        0x00b0:  6f78 2f31 382e 300d 0a41 6363 6570 743a  ox/18.0..Accept:
        0x00c0:  2074 6578 742f 6874 6d6c 2c61 7070 6c69  .text/html,appli
        0x00d0:  6361 7469 6f6e 2f78 6874 6d6c 2b78 6d6c  cation/xhtml+xml
        0x00e0:  2c61 7070 6c69 6361 7469 6f6e 2f78 6d6c  ,application/xml
        0x00f0:  3b71 3d30 2e39 2c2a 2f2a 3b71 3d30 2e38  ;q=0.9,*/*;q=0.8
        0x0100:  0d0a 4163 6365 7074 2d4c 616e 6775 6167  ..Accept-Languag
        0x0110:  653a 2065 6e2d 5553 2c65 6e3b 713d 302e  e:.en-US,en;q=0.
        0x0120:  350d 0a41 6363 6570 742d 456e 636f 6469  5..Accept-Encodi
        0x0130:  6e67 3a20 677a 6970 2c20 6465 666c 6174  ng:.gzip,.deflat
        0x0140:  650d 0a43 6f6e 6e65 6374 696f 6e3a 206b  e..Connection:.k
        0x0150:  6565 702d 616c 6976 650d 0a0d 0a         eep-alive....
```

Listing 1-2: Tcpdump output showing content

Notice how this listing includes much more information than the headers in Listing 1-1. Not only do you see full header information (MAC addresses, IP addresses, IP protocol, and so on), but you also see the higher-level content sent by the web browser. You can read the `GET` request, the user agent, some HyperText Transfer Protocol (HTTP) headers (`Accept`, `Accept-Language`, `Accept-Encoding`, and so on). Although it appears a bit unwieldy in this format, the granularity is undeniable.

Using a Graphical Tool to View the Traffic

We can look at this same full content traffic with a graphical tool like Wireshark (*http://www.wireshark.org/*), as shown in Figure 1-7. Wireshark is an open source protocol analysis suite with a rich set of features and capabilities. In Figure 1-7, I've highlighted the packet showing a `GET` request, corresponding to the same packet depicted in Listing 1-2.

Clearly, if you have access to full content data, there are few limits to the sorts of analysis you can conduct. In fact, if you have all the traffic passing on the wire, you can extract all sorts of useful information.

The next section shows how to assemble packets to capture interactions between computers, including messages and files transferred.

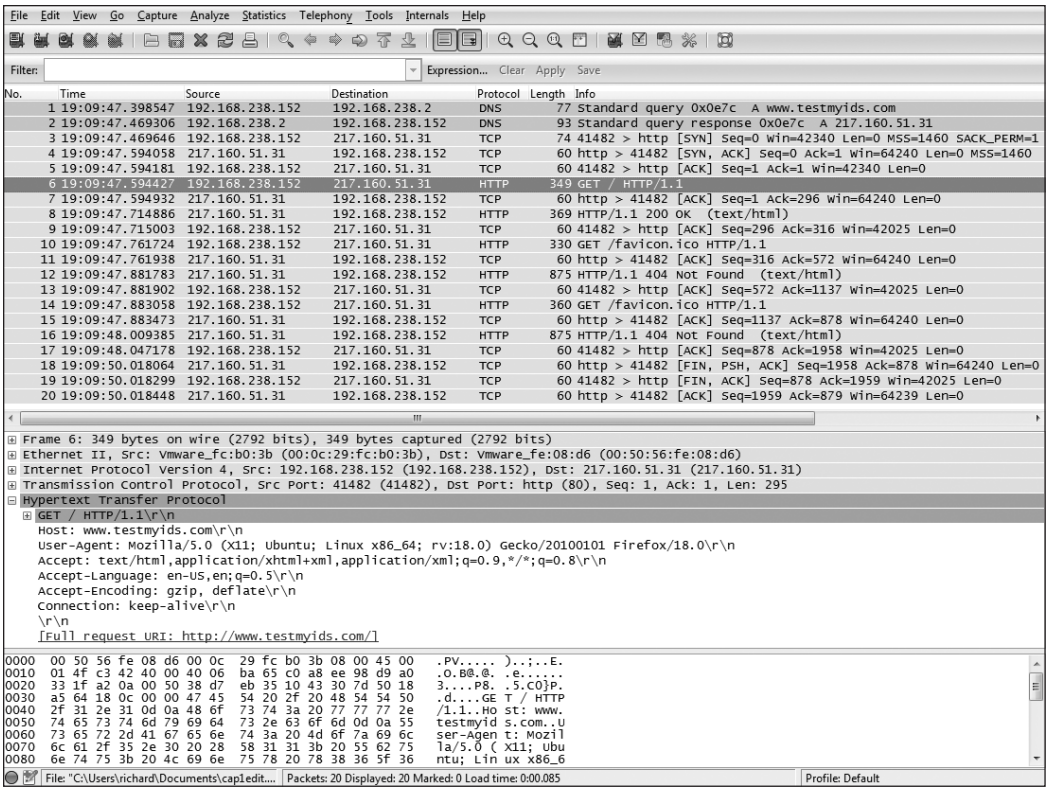

Figure 1-7: Wireshark's rendition of web browsing traffic

Extracted Content Data

Extracted content refers to high-level data streams—such as files, images, and media—transferred between computers. Unlike with full content data, which includes headers from lower levels of the communication process, with extracted content, we don't worry about MAC addresses, IP addresses, IP protocols, and so on. Instead, if two computers exchange a file, we review the file. If a web server transfers a web page to a browser, we review the web page. And, if an intruder transmits a piece of malware or a worm, we review the malware or worm.

Wireshark can depict this content as a stream of data, as shown in Figure 1-8. The `GET` message shows content sent from the web browser to the web server. The `HTTP/1.1` message shows content sent from the web server back to the web browser. (I've truncated the conversation to save space.) Then the web client makes a request (`GET /favicon.ico`), followed by another reply from the web server (`HTTP/1.1 404 Not Found`).

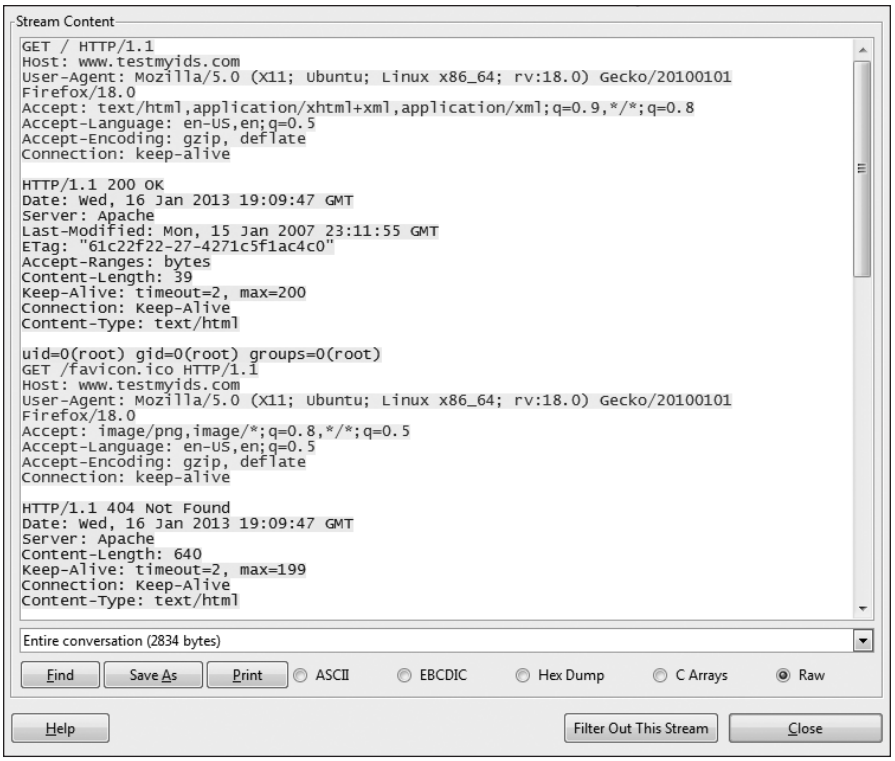

Figure 1-8: Wireshark's rendition of extracted content

When you visit a website, the actions that produce the messages shown in Figure 1-8 are happening behind the scenes to get you the content you want. Security teams can analyze this data for suspicious or malicious content. For example, intruders may have injected links to malicious websites into websites trusted by your users. NSM professionals can find these evil links and then learn if a user suffered a compromise of his computer.

In addition to viewing web browsing activity as text logs or data streams, it can be helpful to see reconstructions of a web browsing session. As you can see in Figure 1-9, the open source tool Xplico (*http://www.xplico.org/*) can rebuild a web page whose content was captured in network form.

Figure 1-9 shows an Xplico case where the analyst chooses to rebuild the *http://www.testmyids.com/* website. With a tool like Xplico, you don't need to look at possibly cryptic messages exchanged by web servers and web browsers. Xplico and other network forensic tools can try to render the website as seen by the user.

For the past several years, NSM practitioners have extracted content from network traffic in order to provide data to other analytical tools and processes. For example, NSM tools can extract executable binaries from network streams. Analysts can save and submit these artifacts to antivirus engines for subsequent analysis. They can also reverse engineer the samples or "detonate" them in a safe environment for deeper examination.

Now we will continue with a new form of NSM data: session data.

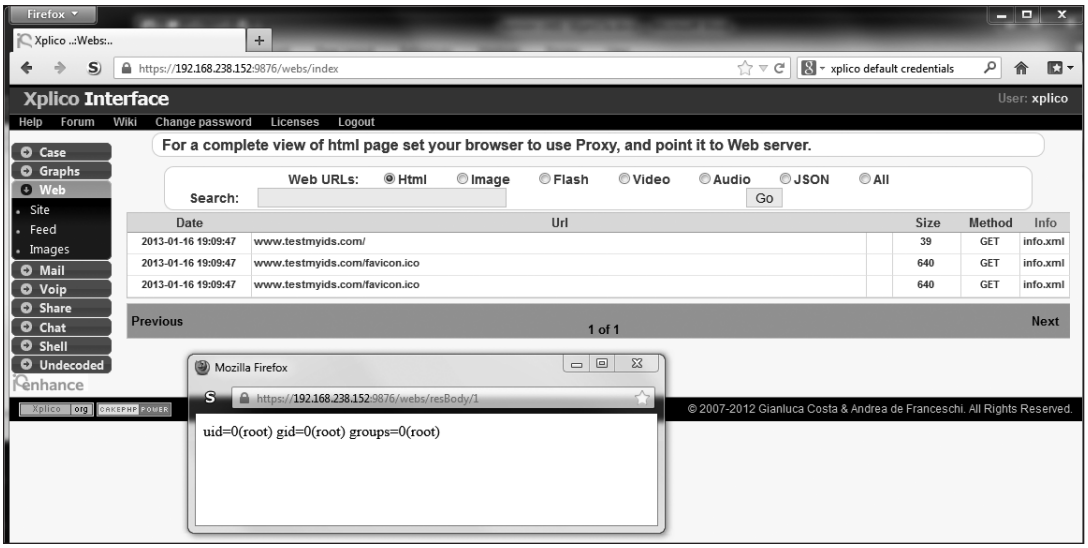

Figure 1-9: Xplico's rendition of the http://www.testmyids.com/ *website*

Session Data

Session data is a record of the conversation between two network nodes. An NSM tool like Bro (*http://www.bro.org/*) can generate many types of logs based on its inspection of network traffic. Listing 1-3 shows an excerpt from the Bro *conn.log* that corresponds to the web browsing activity discussed in "Full Content Data" on page 16.

```
#fields
ts                          uid          id.orig_h         id.orig_p  id.resp_h       id.resp_p
 proto   service  duration     orig_bytes  resp_bytes   conn_state  local_orig  missed_bytes
 history     orig_pkts  orig_ip_bytes   resp_pkts  resp_ip_bytes  tunnel_parents  orig_cc  resp_cc

#types
time                        string       addr             port       addr            port
 enum    string   interval    count       count        string      bool        count
 string  count        count         count         count   table[string]     string     string

2013-01-16T19:09:47+0000❶  90E6goBBSw3  192.168.238.152❷ 41482❸   217.160.51.31❹
 80❺  tcp❻     http     2.548653    877❼       1957❽        SF          T           0
 ShADadfF   9          1257            9          2321          (empty)          -        DE

2013-01-16T19:09:47+0000     49vu9nUQyJf  192.168.238.152  52518     192.168.238.2
 53    udp     dns      0.070759     35          51          SF          T           0
 Dd         1          63             1           79          (empty)          -        -
```

Listing 1-3: Sample session data from the Bro connection log (conn.log)

Session data collapses much of the detail into core elements, including the timestamp ❶, source IP address ❷, source port ❸, destination IP address ❹, destination port ❺, protocol ❻, application bytes sent by the source ❼,

application bytes sent by the destination ❽, and other information. One could generate session data from full content data, but if hard drive space is at a premium, then logging only session data might be a good option.

The open source session data tool Argus (*http://www.qosient.com/argus/*) can also generate records for this traffic, as shown in Listing 1-4.

```
StartTime        Flgs  Proto  SrcAddr         Sport  Dir  DstAddr       Dport
 TotPkts   TotBytes State

19:09:47.398547  e     udp    192.168.238.152.52518  <->  192.168.238.2.53
      2         170   CON

19:09:47.469646  e     tcp    192.168.238.152.41482   ->  217.160.51.31.80
     18        3892   FIN
```

Listing 1-4: Sample session data from Argus

The open source tool Sguil (*http://www.sguil.net/*) can also be used to view session data. Sguil traditionally used the SANCP tool (*http://nsmwiki.org/SANCP*) to collect session data and render it as shown in Figure 1-10.

Sensor	Cnx ID	Start Time	End Time	Src IP	SPort	Dst IP	DP...	Pr	S Pckts	S Bytes	D Pckts	D Bytes
sovm-eth1	5.1358363387000000183	2013-01-16 19:09:47	2013-01-16 19:09:50	192.168.238.152	41482	217.160.51.31	80	6	9	1077	9	2141
sovm-eth1	5.1358363387000000182	2013-01-16 19:09:47	2013-01-16 19:09:47	192.168.238.152	52518	192.168.238.2	53	17	1	43	1	59

Figure 1-10: Sguil's rendition of session data collected by SANCP

Session data tends to focus on the call details of network activity. This information includes who spoke, when, and how, and the amount of information each party exchanged. The nature of those exchanges is not usually stored in session data. For that, we turn to transaction data.

NOTE *Listings 1-3 and 1-4 and Figure 1-10 each show slightly different output. We'll examine why later in the book.*

Transaction Data

Transaction data is similar to session data, except that it focuses on understanding the requests and replies exchanged between two network devices.

We'll use Bro to explore an example of transaction data. As you can see in Listing 1-5, reviewing Bro's *http.log* shows the request and reply between a web browser and web server.

```
2013-01-16T19:09:47+0000        90E6goBBSw3     192.168.238.152 41482   217.160.51.31   80
1       GET❶     www.testmyids.com       /               -         Mozilla/5.0 (X11; Ubuntu;
Linux x86_64;
rv:18.0) Gecko/20100101 Firefox/18.0    0       39      200❹     OK              -       -
-       (empty) -       -       -       text/plain      -       -
```

```
2013-01-16T19:09:47+0000        9OE6goBBSw3     192.168.238.152 41482   217.160.51.31   80
2       GET❷     www.testmyids.com       /favicon.ico    -       Mozilla/5.0 (X11; Ubuntu;
Linux x86_64;
rv:18.0) Gecko/20100101 Firefox/18.0    0       640     404❺     Not Found       -       -
-       (empty) -       -       -       text/html       -       -

2013-01-16T19:09:47+0000        9OE6goBBSw3     192.168.238.152 41482   217.160.51.31   80
3       GET❸     www.testmyids.com       /favicon.ico    -       Mozilla/5.0 (X11; Ubuntu;
Linux x86_64;
rv:18.0) Gecko/20100101 Firefox/18.0    0       640     404❺     Not Found       -       -
-       (empty) -       -       -       text/html       -       -
```

Listing 1-5: Sample transaction data from a Bro HTTP log (http.log)

These records show the web browser's `GET` request for the web root `/` ❶, followed by one request for a *favicon.ico* file ❷, and a second request for a *favicon.ico* file ❸. The web browser responded with a `200 OK` for the web root `GET` request ❹ and two `404 Not Found` responses for the *favicon.ico* file ❺.

This is just the sort of information a security analyst needs in order to understand the communication between the web browser and the web server. It's not as detailed as the full content data, but not as abstract as the session data. Think of it this way: If full content data records every aspect of a phone call, and session data tells you only who spoke and for how long, then transaction data is a middle ground that gives you the gist of the conversation.

Let's briefly look at transaction data for a different aspect of the sample web browsing activity: DNS requests and replies, as shown in Listing 1-6. Again, we don't need all the granularity of the full content data, but the session data would just show that an exchange took place between the two computers. Transaction data gives you a middle ground with some detail, but not an excessive amount.

```
2013-01-16T19:09:47+0000        49vu9nUQyJf     192.168.238.152 52518
192.168.238.2   53      udp     3708    www.testmyids.com       1       C_
INTERNET      1       A       0       NOERROR F       F       T       T
0       217.160.51.31   5.000000
```

Listing 1-6: Sample transaction data from a Bro DNS log (dns.log)

Bro and other NSM tools can render various forms of transaction data, as long as the software understands the protocol being inspected.

You may get the sense that transaction data is the "perfect" form of NSM data; it's not too hot and not too cold. However, each datatype has its uses. I will show why this is true when we look at tools in detail in Chapters 6, 7, and 8, and at case studies in Chapters 10 and 11.

Statistical Data

Statistical data describes the traffic resulting from various aspects of an activity. For example, running the open source tool Capinfos (packaged with Wireshark) against a file containing stored network traffic gives the results shown in Listing 1-7. The example shows key aspects of the stored network traffic, such as the number of bytes in the trace (file size), the amount of actual network data (data size), start and end times, and so on.

```
File name:           cap1edit.pcap
File type:           Wireshark/tcpdump/... - libpcap
File encapsulation:  Ethernet
Packet size limit:   file hdr: 65535 bytes
Number of packets:   20
File size:           4406 bytes
Data size:           4062 bytes
Capture duration:    3 seconds
Start time:          Wed Jan 16 19:09:47 2013
End time:            Wed Jan 16 19:09:50 2013
Data byte rate:      1550.44 bytes/sec
Data bit rate:       12403.52 bits/sec
Average packet size: 203.10 bytes
Average packet rate: 7.63 packets/sec
SHA1:                e053c72f72fd9801d9893c8a266e9bb0bdd1824b
RIPEMD160:           8d55bec02ce3fcb277a27052727d15afba6822cd
MD5:                 7b3ba0ee76b7d3843b14693ccb737105
Strict time order:   True
```

Listing 1-7: Statistical data from Capinfos

This is one example of statistical data, but many other versions can be derived from network traffic.

Wireshark provides several ways to view various forms of statistical data. The first is a simple description of the saved traffic, as shown in Figure 1-11. This figure shows information similar to that found in the Capinfos example in Listing 1-7, except that it's generated within Wireshark.

Wireshark also provides protocol distribution statistics. Figure 1-12 shows traffic broken down by type and percentages.

In Figure 1-12, you can see that the trace consists of all IP version 4 (IPv4) traffic. Within that protocol, most of the activity is Transmission Control Protocol (TCP), at 90 percent. The remaining 10 percent is User Datagram Protocol (UDP). Within the TCP traffic, all is HTTP, and within the UDP traffic, all is DNS. Analysts use these sorts of breakdowns to identify anomalies that could indicate intruder activity.

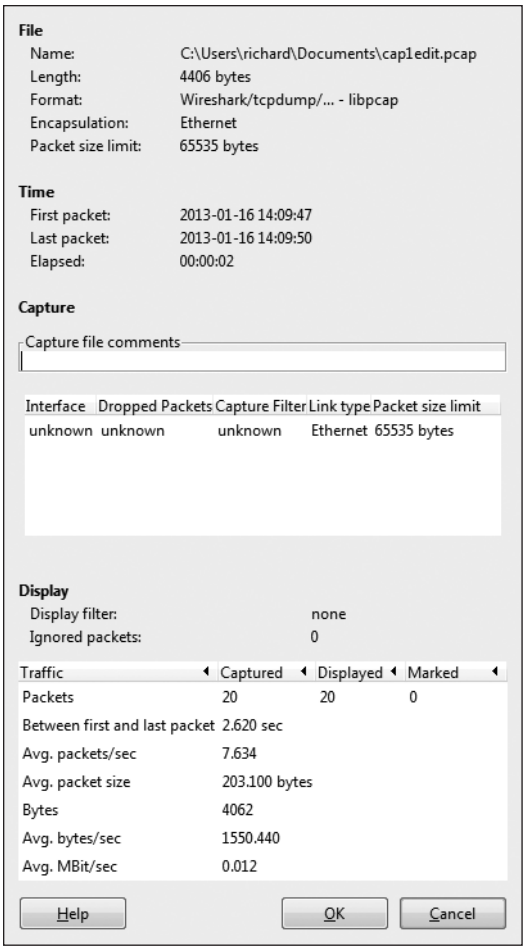

Figure 1-11: Basic Wireshark statistical data

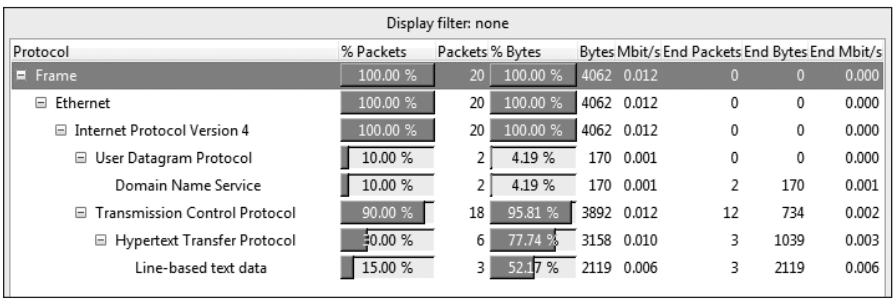

Figure 1-12: Wireshark protocol distribution statistics

Another form of statistical data generated by Wireshark is packet length statistics, as shown in Figure 1-13.

Figure 1-13 shows that the majority of the traffic has packet lengths of 40 to 79 bytes. In some organizations, this could indicate suspicious or malicious activity. For example, an attacker conducting a distributed denial-of-service (DDoS) attack might generate millions of smaller packets to bombard a target. That is not the case here; the packets are mainly 40 to 79 bytes, or 320 to 1279 bytes.

Metadata, discussed next, is related to statistical data, and is just as valuable.

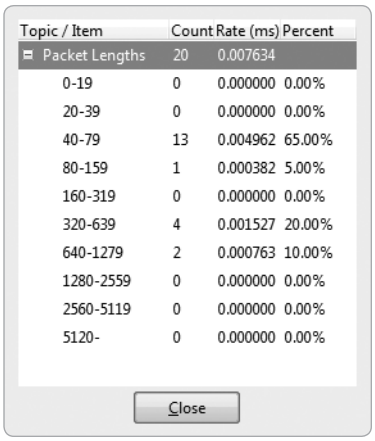

Figure 1-13: Wireshark packet length statistics

Metadata

Metadata is "data about data." In order to generate metadata, we extract key elements from network activity, and then leverage some external tool to understand it. For example, we have seen many IP addresses in the traffic thus far. Who owns them? Does their presence indicate a problem for us? To answer those questions, we could inspect the domains and IP addresses for the traffic and retrieve metadata, beginning with a query of the WHOIS database for IP information, as shown in Listing 1-8.

```
% This is the RIPE Database query service.
% The objects are in RPSL format.
%
% The RIPE Database is subject to Terms and Conditions.
% See http://www.ripe.net/db/support/db-terms-conditions.pdf

% Note: this output has been filtered.
%       To receive output for a database update, use the "-B" flag.

% Information related to '217.160.48.0 - 217.160.63.255'

inetnum:        217.160.48.0 - 217.160.63.255
netname:        SCHLUND-CUSTOMERS
descr:          1&1 Internet AG
descr:          NCC#1999110113
country:        DE
admin-c:        IPAD-RIPE
tech-c:         IPOP-RIPE
remarks:        in case of abuse or spam, please mailto: abuse@oneandone.net
status:         ASSIGNED PA
mnt-by:         AS8560-MNT
source:         RIPE # Filtered

-- snip --
```

```
% Information related to '217.160.0.0/16AS8560'

route:          217.160.0.0/16
descr:          SCHLUND-PA-3
origin:         AS8560
mnt-by:         AS8560-MNT
source:         RIPE # Filtered

% This query was served by the RIPE Database Query Service version 1.50.5
(WHOIS1)
```

Listing 1-8: WHOIS output for IP address

Next, query WHOIS for domain information, as shown in Listing 1-9.

```
   Domain Name: TESTMYIDS.COM
   Registrar: TUCOWS DOMAINS INC.
   Whois Server: whois.tucows.com
   Referral URL: http://domainhelp.opensrs.net
   Name Server: NS59.1AND1.CO.UK
   Name Server: NS60.1AND1.CO.UK
   Status: ok
   Updated Date: 11-aug-2012
   Creation Date: 15-aug-2006
   Expiration Date: 15-aug-2014

>>> Last update of whois database: Wed, 16 Jan 2013 21:53:46 UTC <<<

-- snip --

Registrant:
 Chas Tomlin
 7 Langbar Close
 Southampton, HAMPSHIRE SO19 7JH
 GB

 Domain name: TESTMYIDS.COM

 Administrative Contact:
    Tomlin, Chas  chas.tomlin@net-host.co.uk
    7 Langbar Close
    Southampton, HAMPSHIRE SO19 7JH
    GB
    +44.2380420472
 Technical Contact:
    Ltd, Webfusion  services@123-reg.co.uk
    5 Roundwood Avenue
    Stockley Park
    Uxbridge, Middlesex UB11 1FF
    GB
    +44.8712309525    Fax: +44.8701650437
-- snip --
```

Listing 1-9: WHOIS output for domain

The example in Listing 1-9 shows that the domain *testmyids.com* is registered to a user in Great Britain. This is public information that could prove valuable if we need to better understand the nature of this website.

To understand more about the IP addresses in the examples, we might want to analyze routing data to see how *www.testmyids.com* connects to the Internet. NSM analysts might use routing data to link various suspicious IP addresses to each other. RobTex (http://*www.robtex.com*) offers a free resource to show routing data. Figure 1-14 shows its results for *testmyids.com*.

Figure 1-14 shows how the servers hosting *testmyids.com* relate to their part of the Internet. We see that they ultimately get network connectivity via AS number 8560, on the far right side of the diagram. An *Autonomous System (AS)* is an aggregation of Internet routing prefixes controlled by a network. By understanding this information, NSM analysts might link this site to others on the same AS, or group of systems.

Many other forms of metadata can be derived from network traffic. We conclude this section by looking at the application of threat intelligence to network activity.

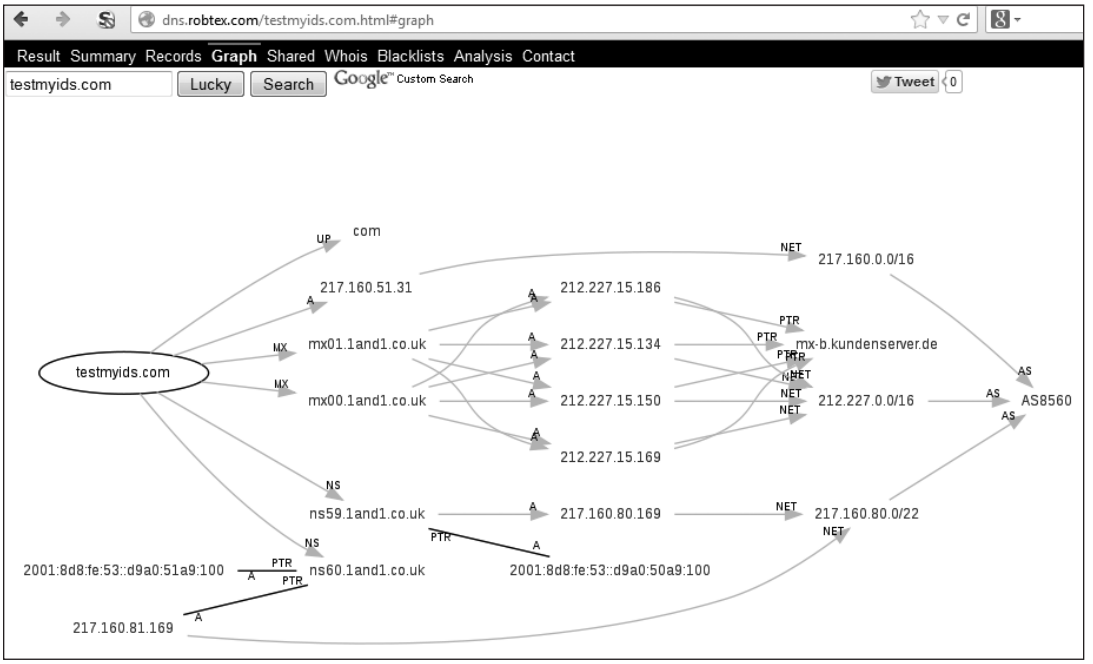

Figure 1-14: Robtex routing information for testmyids.com *domain*

Alert Data

Alert data reflects whether traffic triggers an alert in an NSM tool. An intrusion detection system (IDS) is one source of alert data. Snort (*http://www.snort.org/*) and Suricata (*http://suricata-ids.org/*) are two popular open source IDSs. These tools watch and interpret network traffic, and create a message when they see something they are programmed to report. These

alerts are based on patterns of bytes, or counts of activity, or even more complicated options that look deeply into packets and streams on the wire.

Analysts can review alert data in consoles like Sguil or Snorby (*http://www.snorby.org/*). For example, Figure 1-15 shows a Snorby screen displaying the details of an IDS alert triggered by visiting *http://www.testmyids.com/*, and Figure 1-16 shows what Sguil displays.

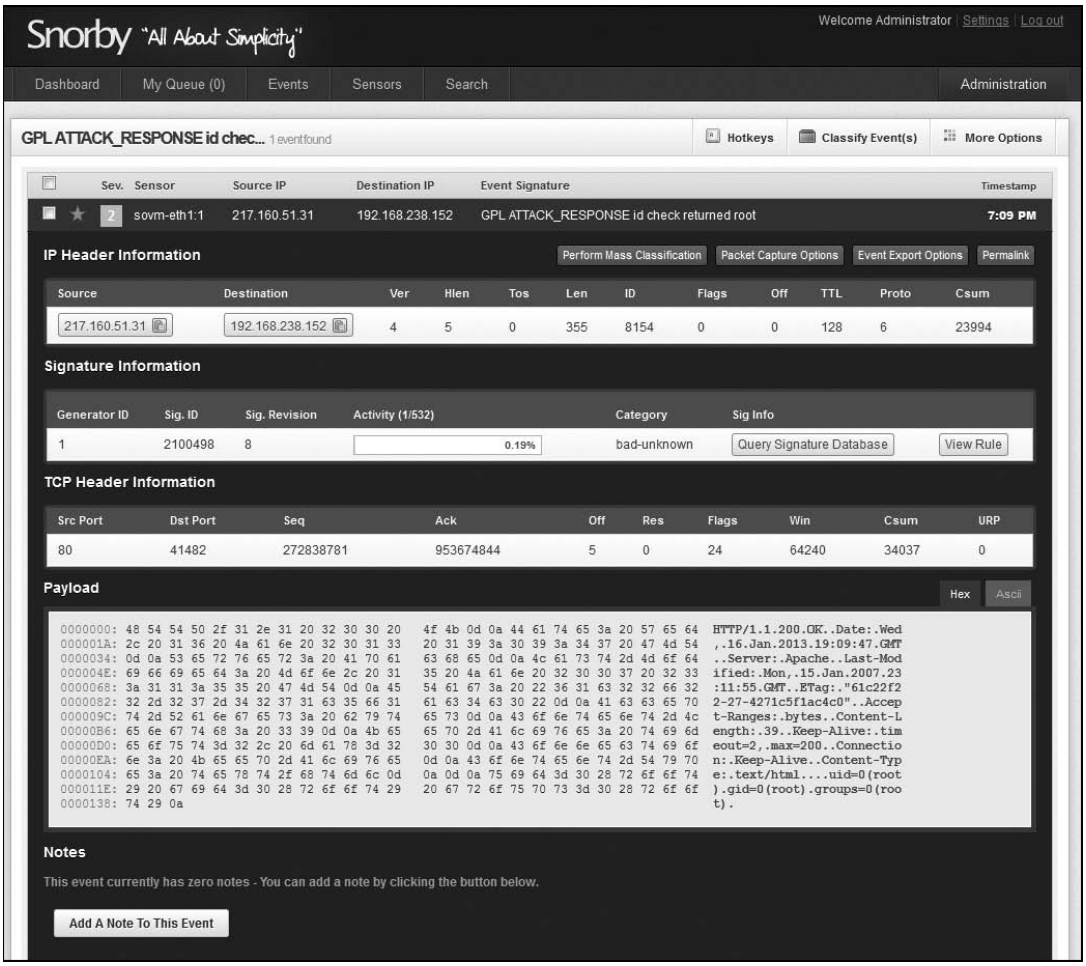

Figure 1-15: Snorby alert data

In a single console, Snorby collects a wealth of information, such as the IP addresses involved with the connection and the packet that generated the alert. Snorby also gives analysts the ability to search for related data and make incident classification and management decisions based on what they see.

Sguil captures much of the same information as shown by Snorby. The difference is that Snorby is a web-based tool, whereas Sguil is a "thick client" that users install on their desktops. Both sorts of NSM tools display alerts by correlating known or suspected malicious data with network activity.

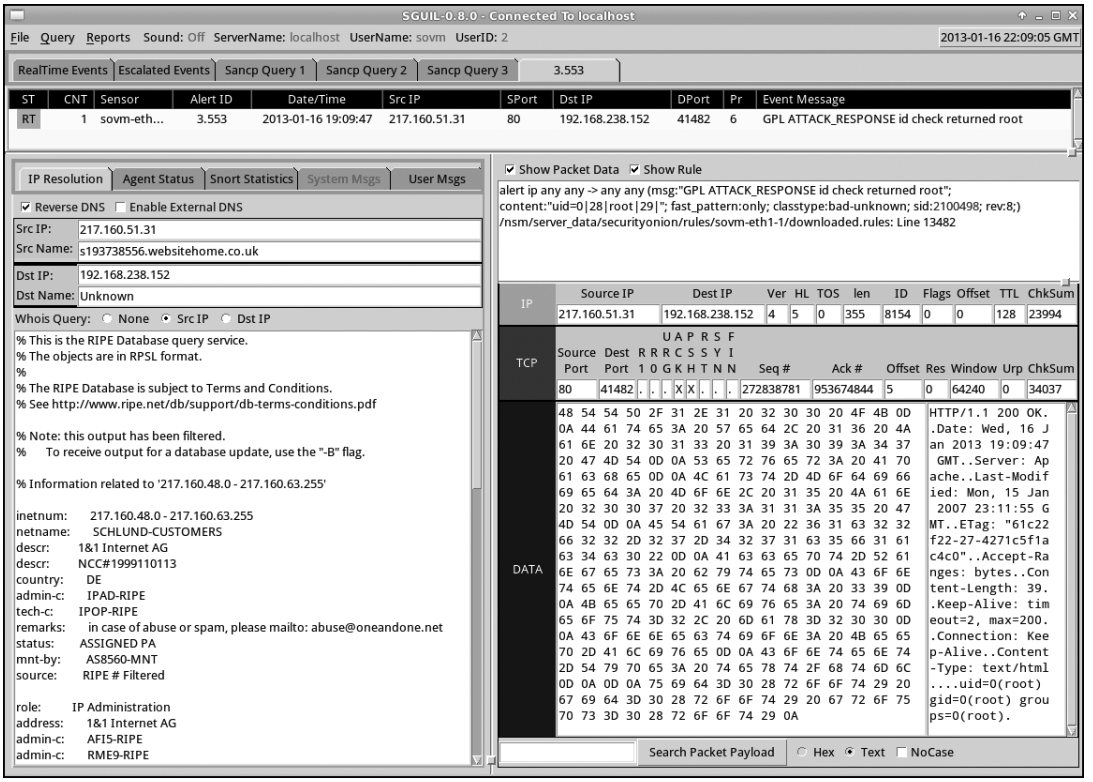

Figure 1-16: Sguil alert data

In the previous examples, the Snort IDS generated `GPL ATTACK_RESPONSE id check returned root` alerts as a result of a user visiting the *http://www.testmyids.com/* website. It's up to the analyst to decide if this is benign, suspicious, or malicious. How to obtain data, use the tools, and operate a process to make this decision is the focus of this book, and I answer these questions in the chapters that follow.

What's the Point of All This Data?

The variety and diversity of NSM data equips CIRTs to detect, respond to, and contain intruders in a manner that complements the efforts of other tools and systems. NSM can make it possible for analysts to discover and act on intrusions early on in the process, and to use *retrospective security analysis (RSA)* to apply newly discovered threat intelligence to previously collected data in hopes of finding intruders who evaded earlier detection. NSM also gives analysts the data they need for *postmortem* analysis, which is an examination following incident resolution.

If I had to leave you with one critical lesson from doing NSM operations, it's this: The best way to use network-centric data to detect and respond to intrusions is to collect, analyze, and escalate as much evidence as your technical, legal, and political constraints allow. This means doing

more than waiting for an IDS to trigger an alert, or beginning to collect more information about an incident only after an IDS triggers an alert. Successful NSM operations are always collecting multiple forms of NSM data, using some of it for matching activities (via IDS and related systems) and hunting activities (via human review of NSM data). (I'll explain these methods in Chapters 9, 10, and 11.)

The most sophisticated intruders know how to evade IDS signatures and traditional analysis. Only by equipping a CIRT's analysts with the full range of NSM data can you have the best chance of using network-centric evidence to foil those sorts of adversaries. NSM data, and analysts who put it to maximum use, has helped organizations of all sizes and complexities counter a wide range of intruders since the technology and methodology evolved in the 1990s. Despite challenges posed by increasing intruder skill, widespread adoption of encryption, and increasing bandwidth, NSM continues to be a scalable and cost-effective security measure.

NSM Drawbacks

It would not be fair to discuss all the positives of the NSM experience without mentioning a few drawbacks. NSM encounters difficulty when faced with one or more of the following situations.

- Network traffic is encrypted, thus denying access to content. When virtual private networks (VPNs) are active, even source and destination IP addresses may be obscured.
- Network architecture, such as heavy and repeated use of network address translation (NAT) technologies, may obscure source and destination IP addresses.
- Highly mobile platforms may never use a segment monitored by the NSM platform, thereby failing to generate traffic that the CIRT can analyze for malicious activity.
- Extreme traffic volume may overwhelm NSM platforms, or at least require more hardware than the CIRT may have anticipated deploying.
- Privacy concerns may limit access to the sorts of traffic required for real NSM effectiveness.

Those are all accurate descriptions, and other drawbacks probably exist. Chapter 2 discusses how to address some of them. However, in the many years since 1998 when I first learned NSM principles, the system has always benefited my network intrusion detection and response work.

Where Can I Buy NSM?

Perhaps by now you're ready to write a check for a vendor who will ship you a shiny "NSM in a box," ready to conquer evil on your network. Unfortunately, there's more to NSM than software and data.

NSM is an operation that also relies on people and processes. The primary purpose of this book is to help you understand NSM and begin an operation as quickly and efficiently as possible.

A secondary purpose of this book is to help you be able to identify NSM operations when you see them. For example, you may find vendors offering "NSM" services, but you aren't sure whether they've just adopted the lingo without actually implementing the operation. Using this book, you can determine whether they're running a real NSM shop.

Where Can I Go for Support or More Information?

There is no international NSM organization, nor any NSM clubs. Perhaps it's time to start one! Additional resources for learning more about NSM include the following:

- The NSM wiki (*http://nsmwiki.org/*), maintained by David Bianco
- The #*snort-gui* Internet Relay Chat (IRC) channel on Freenode
- The Security Onion website (*http://securityonion.blogspot.com/*) and mailing lists (*http://code.google.com/p/security-onion/wiki/MailingLists*)
- Members of the NetworkSecurityMonitoring list on Twitter (*https://twitter.com/taosecurity/networksecuritymonitoring/members*), some of whom also operate blogs (linked from their Twitter profiles)
- My other books on the topic (listed in the preface)

Conclusion

This chapter introduced the principles of NSM. Along the way, we looked at a true case study, discussed how NSM fits into existing architectures and tools, and surveyed various forms of NSM data. You may feel overwhelmed by the introduction of numerous tools, datatypes, and concepts in this chapter. That's why I wrote the rest of this book! After practicing, teaching, and writing about NSM since 1999, I've learned that taking an incremental approach is the best way to get colleagues, students, and readers comfortable with NSM.

My goal has been to give you an overall feel for how NSM differs from other security approaches. NSM is a model for action, with network-derived data at the heart of the operations. NSM recognizes that time is the most important element in security, as demonstrated by the state of South Carolina DoR case study. CIRTs and analysts rely on a variety of NSM datatypes, not just packets captured from the wire.

In the rest of the book, I will help you get a basic NSM operation running. I'll show you where to deploy sensors, how they work, what data they collect and interpret, and how to use that data to find intruders. Let's go!