Global Information Assurance Certification Paper

Building Network Intrusion Detection Systems Using Open Source Software

Introduction:

It seems that everyday the news reports that another organization has had its network security compromised. The threats are legion, stemming from viruses and similar malicious code to automatic remote compromise scripts that can allow an attacker full access to a system within seconds. Network administrators simply do not have the time and resources to devote to the defense of their networks that their attackers have. Over the last few years, various companies have developed software to automatically detect intrusions onto networks. Realizing the limiting factors (mostly cost) of deploying these systems, the open source community has risen to the challenge and offers several lower-cost alternatives. In fact it is now possible to implement and deploy an Intrusion Detection System solely using open source software available on the Internet and only requiring a relatively minimal investment in hardware. Though many other possibilities exist, this paper will focus on using SHADOW and Snort along with some other tools to create a viable intrusion detection system.

Traffic vs. Content Analysis:

In order to determine which tools to build your intrusion detection system with, it will be necessary to know something of the two main approaches to intrusion detection – traffic and content analysis.

Most commercial intrusion detection systems use content analysis. The vendors realize that their target audience is the overworked administrator who has barely enough time in the day to do his or her job let alone review gigabytes of data trying to determine if one of his or her servers is now serving The Mummy Returns to the entire Internet. Content analysis looks for signatures within the packet payload and will respond appropriately when a match is found. This is similar to the way most anti-virus software works. Also like anti-virus software, you need to supply the intrusion detection system with a signature or rules file so that it knows what to look for. It is essential that the administrator take the time to tweak this file so that he is not inundated with too many false positives, but also is not missing vital alerts. Content analysis requires the capture of the entire packet. Ethernet packets can grow to 1500 bytes. With a fairly high-speed connection, this can require large amounts of disk space and significant CPU time to process the data. The systems typically only log the abnormal traffic. The advantages of content analysis are it may be faster and easier to interpret, and close-to-real-time detection is possible. The disadvantages are that false positives and negatives are more common, and it will require more system resources to run.

Through traffic analysis, the interpreter hopes to see patterns in the packet header that may indicate abnormal network behavior. This does require that the analyst be trained to interpret this data and this analysis can be very time consuming. On a positive note, it does not usually take very long for the analyst to become familiar with his or her organization's typical network traffic, and the analysis can become easier over time. Since the analyst is only looking at headers, it is only required to capture the header. The default 68 bytes that tcpdump captures, for example, provides more than enough data to get this information. To get an accurate analysis it is necessary to capture every packet on the wire. Even at 68 bytes, this will consume large amounts of disk space. The main advantage of traffic analysis that is possible to get a more accurate interpretation of the data. The disadvantages are that it requires a trained

analyst to accurately interpret the data, it is not possible to have close-to-real-time detection, and it requires a large amount of disk space.

Hybrid Analysis (Can we combine the two methods?):

A hybrid method that collects the entire packet for processing though a content filter for quick analysis yet also gets every packet on the wire for further review by the interpreter could work well. The main drawback of this method is that to be as accurate as possible, the as much of the packet as possible must be captured and kept. It could require vast amounts of drive space to store this data, but now such a system would allow both close-to-real-time detection capability and the ability to further investigate suspicious activity. This method could also simplify determining how a system was compromised since the data could be played back.

The SHADOW Method:

SHADOW is a system utilizing the CIDER (Cooperative Intrusion Detection Evaluation and Response) concept. CIDER is to utilize common public domain software for anyone to inexpensively protect their systems.

> "A SHADOW system (short for SANS's Heuristic Analysis system for Defensive Online Warfare), can be built using freely-available software and existing hardware that can be purchased for less than $10,000." (Linux Weekly News)

SHADOW is a collection of PERL scripts that interact with tcpdump and SSH to provide a sorted, easy-to-read traffic analysis in the form of an html document that can be accessed by any web browser. SHADOW can run on most UNIX-like systems (though some are more recommended than others).

SHADOW and most other intrusion detection systems use two components – a sensor and an analyzer. With SHADOW, a sensor starts a new tcpdump process every hour (and stopping the previous hour's) and the analyzer, using SSH, pulls the previous hour's file. The analyzer then runs the tcpdump data through a series of tcpdump filters and builds an html page that can be served using Apache. Any browser can now be used to perform traffic analysis for that hour's data. All of this can be run on one of several open source UNIX platforms such as FreeBSD or Linux. This is an excellent example of tying two open source tools together (tcpdump and SSH), an open source development tool (PERL), and an open source operating system to build a functioning intrusion detection system. Since it's inception, there have been several other tools that have come along that could expand on this concept.

Then Along Came Snort:

Snort was born in 1998. Billed as a "Lightweight Intrusion Detection System", it has become very popular with systems administrators recently. By itself Snort is an open source, rules-based, content analysis system. It compiles on most UNIX platforms and is also available for Windows NT/2000 as well. The rules are easy to develop and understand, and author Martin Roesch has built in compatibility with tcpdump binary files. This allows Snort to interoperate with various other tools. The Snort rules have the ability to do close to real-time alerting and response.

Upon visiting the Snort home page (http://www.snort.org/) the surfer will be presented with the latest news and updates. Though a powerful tool by itself, other users have contributed tools to make using

Snort even easier and more powerful. When a user clicks on the Downloads link, he or she will be presented with a list of contributions for each version. Anyone using Snort will probably want to grab a tool to sort and summarize the logs into a readable format. Snort-sort.pl is one such program. This not only performs the sorting task, but also presents the sorted information as an html page. Because of the way in which Snort was designed, perhaps it is possible to develop a working hybrid intrusion detection system.

The Building Blocks (Some of those open source programs mentioned in the title):

So far a few of the open source programs that can be used have been mentioned. It is now time to examine them more closely for their possible uses.

Tcpdump has been mentioned and will be mentioned frequently. Tcpdump is a network sniffer. Running on a computer as a traffic sensor, tcpdump will see and collect all traffic on the wire. Although there are several other sniffers available, tcpdump has some useful advantages. First tcpdump has been ported to just about every platform so it is readily available. Second, several other network utilities can read and process tcpdump output. However, tcpdump does require that libpcap library be installed on the system. Both of these can be found at ftp://ftp.ee.lbl.gov/.

Logsurfer is a tool for monitoring text log files for anomalous events in real-time. It can send messages when a rule is matched so that an administrator can react quickly to an event. Similar to SWATCH, Logsurfer provides more features and does not require PERL to run. Logsurfer can be acquired at http://www.cert.dfn.de/eng/logsurf/.

SHADOW as a package, is an open source intrusion detection system available from http://www.nswc.navy.mil/ISSEC/CID/step.tar.gz. SHADOW is perfectly usable by itself or the scripts can be modified to drive another intrusion detection system

Snort is an open source intrusion detection system available from http://www.snort.org/. Snort, written in C, is a stand-alone program that can be easily modified, but it's true power becomes apparent when it is complemented with other open source software and may lead to a very robust intrusion detection system.

SSH (Secure shell) is freely available as OpenSSH. The SSH suite replaces the "r" services (i.e. rsh, rlogin, rexec) with a set of utilities having all of the same functions, but which communicate over an encrypted channel. SSH should be used for all communications between computers whenever possible. OpenSSH is available at http://www.openSSH.org/.

Apache is an open source web server and if they are to believed the most widely used web server on the Internet. Apache runs on several Unix platforms as well as Windows NT and 2000. Apache can be acquired from http://www.apache.org/.

Linux is an open source Unix-like operating system. There are several distributions available, some more secure than others. Red Hat (http://www.redhat.com/) has been popular because of its ease to set up and configure. It is important to remember that whatever intrusion detection system is deployed; it will only be as secure as the operating system it resides on. A default configuration will not be acceptable. It should be noted that dropped packets would not be detected on an intrusion detection system running under Linux.

Net/Free/OpenBSD are popular open source versions of the Berkeley Unix. These are becoming used more and more frequently as the default configurations do not have "everything" turned on; this is a problem with several distributions of Linux. The BSD's do not have the same problem with dropped packets as Linux.

Obviously these are but a few of the tools available to an administrator wishing build an intrusion detection system. The tools section of the Security Focus website, the home of Bugtraq, is an excellent source for more resources. (http://www.securityfocus.com/).

Some of The Possibilities:

The first method to explore would be a SHADOW/Snort hybrid. The below example illustrates how SHADOW may drive the process while Snort and other programs processes the data.

SHADOW includes some useful PERL scripts for managing the collection and processing of data from the sensor. Snort can process tcpdump binary files. With little effort, the SHADOW code can be modified to process the tcpdump data through Snort and then onto a post processor/formatter program such as snort-sort.pl. One caveat is that the snapshot length (snaplen), the amount of data taken from each packet) for tcpdump on the sensor will have to be modified so that more than 68 bytes are captured. Because of the storage issues involved in capturing entire packets, the administrator will have to compromise on a snaplen that will give the most useful information without overwhelming his or her storage capacity and processing time. This method will provide the administrator with the most data for forensic analysis.

This method works well when there is limited manpower and time to analyze logs. Data will not be analyzed in real-time with this method, but it is unlikely a single administrator would be able to respond immediately on a 24/7 basis. A large, fast drive array would be recommended. RAID 0 though not fault tolerant will provide the fastest reads and writes. This will be important, as there is only an hour to process the data.

To summarize this method:

1. The sensor launches tcpdump with a longer snaplen every hour.
2. Every hour the analyzer makes a secure connection via SSH to the sensor to pull down the previous hour's data.
3. The analyzer runs Snort, processing the tcpdump data and generates an alert file.
4. Another program processes the alert file into a sorted more readable form for display by the SHADOW web interface.

Software needed (all is open source and freely available):

- SHADOW
- Tcpdump
- SSH
- Snort
- Snort-sort.pl

- Apache web server
- A UNIX (Solaris, Linux, FreeBSD, NetBSD etc.)

Snort has great potential for a real-time intrusion detection system as well. Snort can be run on a lone sensor in conjunction with a log watcher program such as swatch. Snort will generate alerts and swatch can be configured to do email notifications of alerts. A log rotator will be necessary as well to keep the alert files organized. A feature of Snort that has not been mentioned yet is its ability to provide a response to an alert. A rule can be configured to send TCP resets to either or both sides of a connection when a rule is matched. Several commercial intrusion detection systems have similar capabilities. This feature should be used with caution, as the potential for denial of service attacks is great.

To summarize this method:

1. Snort runs in real-time on a host located on the network.
2. A log watcher runs alerting the administrator of any alerts Snort generates.
3. A log rotator manages the logs.

Software needed (all is open source and freely available):

- Snort
- Swatch
- A UNIX (Linux, FreeBSD, NetBSD etc.)
- An email program (usually included in of the above UNIX's)

This is a simple method that can inform an administrator of a problem in close to real-time. This would work best with a security staff that can be available 24x7 and can investigate alerts and determine if they are false positives or actual incidents. The Snort real-time response would be more useful in this situation. It could be used to shut down a connection while the alert is investigated and then more permanently acted upon either by allowing the connection or adding a rule to the firewall. If running an Ipchains firewall, this could be taken a step further. Guardian one of the companion tools for Snort will dynamically update Ipchains firewall rules based on Snort generated alerts.

A Final Caveat:

It should be noted that if any such system, bandwidth is also be a limiting factor. These intrusion detection systems should be able to handle at least 10mbs. Some commercial systems claim to be able to handle 100mbs. The maximum manageable throughput will ultimately be affected by the speed of the sensor's processor and its disk regardless of the efficiency of the code. A fast disk array will go a long way in logging large amounts of traffic without loss.

In Conclusion:

This paper described easy-to-implement examples of open source intrusion detection systems. However, it barely scratched the surface of what is available to a creative administrator. The options are limitless. Existing programs can be tied together using simple scripting languages and scheduled tasks. A more ambitious administrator can modify the source code directly to fit his or her needs and even take

these concepts a step further to provide for host based intrusion detection and auditing. It is not necessary to spend $50,000 dollars for a commercial intrusion detection system and then be burdened with high, annual maintenance fees. Intrusion detection systems can be simple and elegant, or large complicated. While no intrusion detection systems will catch every attack and all systems have limitations, hopefully a systems administrator armed with this information will be able to keep his or her organization out of the headlines.

Bibliography

Linux Weekly News, 8 Sep 1998, URL: http://www.lwn.net/1998/0910/shadow.html

Roesch, Martn. "Snort – Lightweight Intrusion Detection for Networks", URL: http://www.snort.org/lisapaper.txt

Van Jacobson, Craig Leres and Steven McCanne, Tcpdump man page, URL: ftp://ftp.ee.lbl.gov/tcpdump.tar.Z (the man pages are included with the source)

Ley, Wolfgang, Logsufer Home Page, URL: http://www.cert.dfn.de/eng/logsurf/

Apache Home Page, URL http://www.apache.org/