Global Information Assurance Certification Paper

Jared McLaren
GSEC v1.3
Secure Open-Source Network IDS

Abstract

Intrusion Detection Systems (IDS) have become an important piece of the defense in depth method used by private corporations, the public sector, and home users alike. It's the next logical piece to add to your network once a firewall is in place. An IDS is a proactive approach to watching the doors that open to your network to see who is rattling the door handle. IDS is a cyber equivalent to a security camera system; it can warn you about potential attacks, but many times can do very little to stop them. Though current IDS solutions are not reliable in stopping attacks, they are valuable information systems. An attacker would love to either bypass or control your IDS to go unnoticed into your network. A compromise of your IDS would effectively eliminate your watchdog sitting at the front door of your network. This is why a secure architecture and deployment plan is so necessary for a network-based IDS.

Many people have been deploying intrusion detection systems in their networks without a thought of creating a secure deployment plan. I am going to offer some help in creating a secure network-based IDS from the ground up using open-source software. Open-source IDS solutions have proven to be just as effective as high-cost commercial products and are obviously quite a bit more affordable and are also much more flexible and scalable. This is not meant to be a how-to guide, but more of an overview of a combination of products I have found effective in my own experience and an explanation of how they create a defense in depth model of security. The advice offered will be a good start to your own plan of developing and deploying a secure and affordable network-based IDS solution.

System Definition

First, an explanation is needed to show the high-level view of how the network-based system works. There is one main central server that will be used for data storage, data reporting, and IDS sensor management to the IDS administrator. I will refer to this central server as the IDS Director. The IDS sensors are all the worker bees of the operation. They will be placed at strategic choke points on your network to monitor traffic and generate alerts when they see a known pattern of malicious data. These alerts are sent over the network back to the IDS Director for storage and further analysis. One IDS Director can handle many IDS sensors at a time. This gives the ability to expand the IDS coverage of your network without having to constantly buy more hardware for IDS Directors.

Network Setup

Now comes the most important part to keeping your IDS secure; your network setup. First of all, each computer that will be used for an IDS sensor should have 2 network interface cards (NIC) installed. The first NIC will be used for monitoring network traffic. This network card will have no TCP/IP stack installed to make it virtually invisible to anyone poking around on your network. The second NIC will be used as your reporting interface that will allow network-based management of your IDS sensor and send alerts to your IDS Director. This reporting interface should be plugged into its own network segment that can be made private from all other nodes on your network. For example, by using Cisco switches one can create a private VLAN for the reporting interfaces of your IDS sensors and your IDS Director. This VLAN can then use access control lists (ACL's) to lock out unauthorized access. A private communications network for your IDS is a very important building block in looking at security from the ground up. The 2 NIC setup and a private VLAN play a major role in securing your sensitive IDS network from the rest of your user network.

Software Overview

The software is the most versatile part of the setup. There are many ways to get the same job done, and this paper is by no means trying to state that this is the best way or the only way to set up an effective IDS solution. Take some time to research what's new and what's effective for software. I'm going to cover open-source software that is just as, if not more, effective as anything on the market. If you're uncomfortable with open-source, then find a commercial solution and use this paper as a guide to knowing what kind of protection measures can be implemented in any network-based IDS. And always, do software research because we know that the way technology goes, as soon as this paper is completed it will be out of date.

Operating System

The operating systems available for use are quite vast, but I'll focus on two. OpenBSD UNIX and RedHat Linux are the systems that I have used the most in my own experiences. I will concentrate on RedHat for the majority of this paper.

OpenBSD is available for free download at http://www.openbsd.org/ and comes with all the basics. Its best features to offer are security and excellent packet sniffing. OpenBSD is a very secure operating system straight out of its installation. There has not been a remote vulnerability in the default OpenBSD installation in over 4 years! It is also a great platform for sniffing network traffic. It outperforms Linux in this area since the packet capturing capability is built into

the operating system. OpenBSD is an excellent choice and is the preferred operating system for raw packet capturing power and security.

A look into the OpenBSD security auditing shows why they are such a stable and secure operating system. Their security auditing team usually has between 6 and 12 dedicated employees following file-by-file analysis of the entire operating system. This is in search of mainly software bugs, but comes up with security holes as well. Most times, the OpenBSD developers find bugs before the hackers too. When a security bug is found, they believe in a full disclosure process to let everyone know all details of the problem. They also strive to be "Secure by Default". All non-essential or potentially dangerous services are disabled by default and wait for the system administrator to take the time to learn about new services and enable them. The secure by default idea is another step that OpenBSD has taken to be one of the most secure operating systems available. ***(1)***

RedHat Linux is a pleasant and easy UNIX clone operating system for use with IDS. RedHat Linux can be downloaded at http://www.redhat.com and comes with just about all the software you need. RedHat has the advantage of being user friendly and is highly supported in the Internet community. It can also be locked down quite easily and comes with many products to make it a secure platform. Its packet capturing performance doesn't compare to the efficiency of OpenBSD, but the libpcap library used by Linux for packet capture is still very effective. For the beginner, I recommend Linux over OpenBSD.

IDS Software: Snort

For the open-source IDS solution, it's hard to beat Snort. Snort is available for free download from http://www.Snort.org and is the shining IDS star of the open-source community. It is a very efficient lightweight IDS product that is extremely scalable and can perform just as well as any commercial IDS on the market. Snort's performance is so efficient that it can run on top of a *nix platform and be effective on an original Pentium-class computer or even a 486. This makes it a very powerful engine to run on IDS sensors.

There are three main subsystems in the Snort binary. There is the packet decoder, the detection engine, and the logging/alerting subsystem. Each is described below.

The packet-decoding engine received TCP/IP information from the network and breaks down and organizes each OSI layer until it works its way to the application layer. Speed is emphasized in this area of Snort to keep up with high-speed networks. Current supported network architecture includes Ethernet, SLIP, and raw data-link protocols with ATM under development. ***(2)***

The detection engine organizes the current rules to make things as efficient as possible. A Chain Header is an attribute developed by the detection engine to organize groups of rules together that all share the same IP and port information on an alert. These are all then linked to Chain Options, which carry the more specific alert information. This grouping of the rules helps Snort's speed when running packet captures against large rule sets. ***(2)***

The last big subsystem of Snort is the logging/alerting module. This module makes Snort extremely versatile. Alerts can be generated in brief format, tcpdump format, or human readable format. Beyond format, there are many different supported systems for Snort to output its data. Snort has support for flat file output, many databases like MySQL, MSSQL, and Postgresql, XML, SNMP, WinPopup alerts, Syslog, UNIX sockets, and more. As you can see, these output abilities make Snort fit into just about any environment. ***(2)***

Snort's open architecture allows for easy implementation of preprocessors (plugins) to the program. There are preprocessors in Snort that help detect portscans, normalize encoded traffic, reassemble fragmented packets, and many more. This open architecture shows how Snort can be very expandable in its capabilities to anyone with programming knowledge.

Database Software: MySQL

MySQL is the database of preference to most Snort users. This will be the database server used by most people to store the alerts generated by IDS sensors. This database server resides on the IDS Director. MySQL is another free and affective product available for download at http://www.mysql.org. It is an open-source project that works on many platforms, including OpenBSD and Linux. MySQL even comes on the RedHat cd and can be installed straight from the RedHat installation program. When putting MySQL into your production environment, it's a good idea to check the MySQL homepage for software patches or updates.

Once MySQL is installed, there are some basic steps that need to be taken to help secure the database system. By default on RedHat, MySQL runs as the mysql user. Be sure to check that it really is running under the mysql user, and not root. It's a good idea to remove the test database that came installed with MySQL. The test database will serve not real purpose for your use and it is not a good idea to leave it there unattended. Next, disallow all remote access to your database. This can be added back in as necessary when IDS sensors come online. Then delete all accounts from the database that have no username. This will now require users to authenticate to MySQL. After this, be sure to assign a password to the root MySQL account. Finally, flush the privileges on the

database to ensure that all new settings are online. **(3)**

IDS Management Software: Apache and PureSecure

The next set of products is my favorite part of the network-based IDS solution I have chosen to write about. This is the set of tools from Demarc known as PureSecure running in coordination with the Internet's flagship web server Apache. PureSecure has both a client piece and a server piece that add extremely valuable functionality to your IDS solution. I will explain all three parts below.

The Apache web server is a free server available for download at http://www.apache.org and is the most popular web server on the Internet. A recent NetCraft survey shows that over 56% of the web sites on the Internet are running on top of Apache. **(4)** It is a very lean and powerful server that is known for its secure status over Microsoft's IIS web server. Apache comes with the RedHat cd and can also be installed from the RedHat installation program. When installing apache, be sure that the service runs as a non-root user. It's also a good idea to go through Apache's configuration file and disable unnecessary modules that are loaded by default. The adventurous administrator may do research on setting up apache in a chroot'ed environment (chroot is a UNIX command). A chroot'ed Apache environment would ensure that even if Apache were to be compromised, an attacker could not affect the operating system. Apache will run on the IDS Director to allow the Demarc PureSecure web front-end to operate.

The Demarc PureSecure products are an outstanding set of tools available for download at http://www.demarc.org and are not for free use in all cases. Read their license agreement to see if you can use it free of charge. The Demarc products run on any platform capable of running perl and integrate with Snort and MySQL. Demarc's PureSecure server, which resides on the IDS Director running from Apache via SSL, is a perl script that runs a web interface to your IDS Director's MySQL database of alerts. It runs an outstanding reporting console that will show you who is attacking your network, what they are doing, and whom they are attacking. PureSecure server has extensive searching features to sift through your IDS alerts for all kinds of details. It also has a feature to email administrators if certain IDS alerts occur or if something goes wrong with the system. The web-based console makes it very easy to manage multiple IDS sensors from one central location. One main advantage of PureSecure server beyond its reporting capabilities is the ability to manage IDS sensor rules remotely. Administrators can change Snort rules remotely on the fly that will then be synchronized with the IDS sensor. This makes the sensors extremely manageable from the IDS Director.

Demarc's PureSecure client resides on each IDS sensor. The client piece makes sure that Snort is always running and keeps checking with the IDS Director to see if there are any rules changes to synchronize. It has fantastic extended features like file integrity checking and PC health monitoring. The file integrity checks can be set up from the PureSecure server web interface and they allow the PureSecure client to check critical system files on the IDS sensor to see if they have been modified. The client uses MD5 checksums to check each file. These checks are important because if a hacker were successful in breaking into your IDS Sensor, their intrusion would be discovered by these integrity checks. The PC Health monitors can check things like available disk space and processor usage. These can also be set up on the PureSecure server web interface and the client piece will then report back its findings. **(5)**

Security Utilities

Now that the main pieces of the IDS have been covered, it's time to discuss utilities that further defend the security of your system. There has to be a way to send data across the network and administer your servers in a way that someone sniffing the network traffic can't gain any valuable information. Administrators also want to assure that the systems are still secure in the case that someone does get a hold of sensitive information. This brings us to products like Stunnel, SSH, IPchains, TCP wrappers, and others.

Stunnel is a product (available for free download at http://www.stunnel.org) ***(11)*** that creates a secure line of communications in an unsecured environment. Stunnel can create a 128-bit SSL encrypted communications line that is transparent to the application using it. For our situation, we will use stunnel to encrypt MySQL traffic from the IDS sensors to the IDS Director. By default, the alerts generated by the IDS sensors are in clear text. This can be dangerous because anyone sniffing network traffic can potentially see your IDS sensors "talking" to the IDS Director. To use stunnel, we have to set up a stunnel listener on the IDS Director that listens for communication from your IDS sensors. Each sensor then has stunnel set up in client mode to encrypt the outgoing alert traffic en route to the IDS Director. Keep in mind that stunnel uses TCP wrappers, which I'll explain shortly. This is a very simplified explanation of how stunnel works, so I recommend researching stunnel a bit more before diving in. The important thing to know is that stunnel will encrypt traffic for you without interfering with MySQL's functionality.

SSL is the protocol that keeps stunnel secure. It's a widely used standard for secure communications over the Internet used mostly in web content. The goals of SSL are to create cryptographic security, interoperability, extensibility, and relative efficiency. ***(6)*** SSL uses a key exchange method to negotiate an encryption algorithm and cryptographic set of keys before any data is even sent.

It's quite a reliable and secure method of communication.

SSH is a utility used for remote administration and file transfer that runs encrypted over a network. SSH, simply put, is like an encrypted version of telnet. The most popular SSH is OpenSSH available for download at http://www.openssh.org/. This is a secure way to remotely administer IDS Sensors and your IDS Director without worrying about malicious users sniffing network traffic and snooping in on your data. File transfers can be performed securely over SSH with the 'scp' command and also with 'sftp' (if enabled on the system). Each IDS sensor as well as the IDS Director should have SSH server running. Simply use a SSH client to connect and communicate securely.

SSH protocol version 1 stays secure by using various cryptographic keys. ***(7)*** Each SSH server first has its own 1024 bit host-specific RSA key. This is used for host identity. When the server starts, a 768 bit RSA server key is generated. This key is then re-created every hour and is never stored on the hard drive. When a client connects to the SSH server, it creates a random 256 bit key that is then encrypted with the host key and the server key to create a random number used as the session key. The rest of the SSH session is then encrypted using conventional ciphers like 3DES or Blowfish.

IPchains is a built-in Linux firewall that can be used to effectively lock down communications between hosts. This also comes with the RedHat cd and can be installed from the RedHat installation program. IPchains is extremely powerful in creating very granular control in network communications. For example, someone only requiring access for reviewing IDS alerts from the PureSecure SSL web interface can be locked down in IPchains to only be able to access port 443 (SSL) traffic on the IDS Director. IPchains is great for creating rules to separate Administration duties. You may create a rule to only allow a limited number of people port 22 (SSH) traffic for remote administration and allow a different group of people port 3306 (MySQL) traffic for database administration. The best way to set up IPchains in this network-based IDS environment is to allow administrators SSH traffic to the IDS Director, MySQL traffic from the IDS sensors to the IDS Director, and allow SSH traffic to the IDS sensors only from the IDS Director. This way the IDS sensors only have to trust and talk to the IDS Director. It will make administrators SSH into the IDS Director before being able to communicate directly with an IDS sensor. This is another line of defense that makes a total compromise of your network-based IDS much more difficult for an attacker. IPchains should also be configured to not allow the IDS sensors to talk to each other just in case an IDS sensor is compromised.

TCP wrappers are used to further lock down the ability of hosts to connect to certain system services. Certain services using TCP wrappers reference a 'hosts.allow' and 'hosts.deny' file that reside in the /etc/ folder on *nix systems.

The names of the files are self-explanatory. The hosts.deny file should always be set up with the entry ALL:ALL. This means all access is denied by default and then anyone needing access to the system must be explicitly added in the hosts.allow file. This is yet another defense in depth mechanism used to control who has access to your IDS Director and your IDS sensors. **(8)**

Other open-source products can be used to further harden your operating system and also test for any network-based or host-based vulnerabilities you might have missed. Bastille is a great operating system hardening tool for Linux. It's available for free download at http://www.bastille-linux.org. This locks down extensive features of Linux like users, services, setuid programs, and other areas of security interest. I highly recommend using this utility to help create a very secure installation of Linux. Once you have completed setting up your system, it would be a good idea to run a vulnerability scanner against your IDS Director and IDS sensors. Nessus is a fantastic network vulnerability scanner available free of charge at http://www.nessus.org. Nessus rivals the quality of high-end commercial vulnerability scanners and remains one of the favorite open-source tools on the Internet.

Conclusion

Defense in depth should be a common thought to you by now. An Intrusion Detection System is a very important system to keep secure due to the sensitive nature of the data it holds. This paper has given a good base to think about when planning for implementation of your own network-based IDS. Start from the ground up with a secure and private network segment. Try using Snort on an operating system hardened by Bastille. Administer that system using a secure SSH connection. Use an encrypted communications line with stunnel to report IDS alerts to your IDS Director. Lock down communications between hosts with IPchains and TCP wrappers. Test out Demarc PureSecure's features of PC health monitoring and file integrity checking and run it through a good web server like Apache. After your system is set up, run Nessus against it all to be sure you didn't miss anything. All of these methods put together reinforce the security of one another. With a focus on security from start to finish, you can rest assured that your IDS data will be as safe as possible.

References:

1) OpenBSD Security Process, Version 1.201. March 19, 2002.
URL: http://www.openbsd.org/security.html

2) Martin Roesch, "Snort – Lightweight Intrusion Detection for Networks", November 1999.
URL: http://www.snort.org/docs/lisapaper.txt

3) Securing MySQL, March 2002.
URL: http://www.canowhoopass.com/weav/wssig/nusphere/securing-mysql.php

4) NetCraft Web server survey, February 2002.
URL: http://www.netcraft.com/survey/

5) Demarc User Guide HTML Version, March 2002.
URL: http://demarc.com/documentation/userguide.html

6) Freier, Karlton, Kocher, "The SSL Protocol v3.0", March 1996.
URL: http://home.netscape.com/eng/ssl3/ssl-toc.html

7) Manual pages: sshd, September 25, 1999.
URL: http://www.openbsd.org/cgi-bin/man.cgi?query=sshd

8) The Official Red Hat Linux Customization Guide, "TCP Wrappers", 2001.
URL: http://www.redhat.com/docs/manuals/linux/RHL-7.2-Manual/custom-guide/tcp-wrappers.html