Passive Vulnerability Detection

"Techniques to passively find network security vulnerabilities"

Ron Gula

rgula@securitywizards.com

September 9, 1999

Detecting network vulnerabilities can be accomplished through passive network monitoring and analysis of the captured data. The strengths and weaknesses of passive monitoring will be discussed as well as a direct comparison to active vulnerability detection techniques. The Dragon intrusion detection system will be used to identify vulnerable network components in real world examples.

Introduction

Network vulnerability analysis is usually conducted in one of two ways. In the first way, a previously known inventory of network systems is manually analyzed for vulnerabilities. Such an analysis may make use of CERT Advisories, network security mailing lists and many other sources of network security information. The analysis simply searches for known vulnerabilities that may be present in known network systems. This method depends on a audit of known systems. The second method employs active discovery of network components, including topologies and the services offered by each component. Discovered network nodes can be actively interrogated for a wide variety of vulnerabilities. Many tools include vulnerability databases that are used to check network nodes for security problems. A wide range of open source and commercial tools are available for this type of testing.

A third technique that only uses passive network monitoring can also be used to identify vulnerabilities. This technique collects network traffic and then compares it against a database of vulnerable system signatures. Only systems that have active network sessions can be monitored with this technique, unless an external stimulus is used.

The Dragon intrusion detection system from Network Security Wizards will be used to demonstrate several different passive vulnerability collection techniques. Dragon has four libraries of detection signatures that search for network attacks, compromises, misuse and passive vulnerabilities. This paper will focus on Dragon's passive vulnerability detection techniques.

Passive Vulnerabilities

We will consider a passive vulnerability as any vulnerability that may be detected simply through analysis of network traffic. Network traffic can be collected by a network monitor, intrusion detection system or any other network device that may selectively search and store network activity. Choosing which network activity to search for in order to identify passive vulnerabilities is the most complex part of this process. We will now consider several types of passive vulnerability detection techniques.

Server Based Vulnerabilities

Security problems known to exist in specific types of network servers may be detected by searching for activity of those network servers. Put another way, this assumption simply states that if "server XYZ" has a security problem, then it may be interesting for a particular network to search for evidence of "server XYZ". It's important to understand each network's topology and composition in order to select likely vulnerable servers to search for. For example, if an organization has never purchased HP-UX servers, it may not be prudent to search for HP-UX related vulnerabilities.

Many network servers have unique strings of ASCII and binary data that exist on unique ports. Signatures to detect these are trivially written. For example, let's consider writing a Dragon signature to detect very old versions of Sendmail, specifically version 4.1 from Sun.

A short discussion of Dragon signatures is in order. Dragon signatures look for specific patterns in network streams. If the data is present in a large packet, then Dragon can search all or parts of the packet. If the packet is small, then the network stream will be reconstructed so that the pattern may be searched. Dragon also has a notion of protected networks or networks of interest. This may be one or more sets of specified CIDR blocks. All packets can be classified as moving to, moving from, internal traffic or external traffic. Each packet must fit into one of those categories. Dragon signatures can also specify searches that originate from a port or are inbound to a port. Here is a Dragon signature to detect Sun's version 4.1 of Sendmail:

T S F B 0 100 25 SENDMAIL:4.1 Sendmail/204.1

Reading from left to right, "T" signifies to search for TCP traffic and "S" specifies to look for patterns in server responses. These responses will have a source port specified by a later argument. "F" signifies to only look at IP packets that are leaving a protected network. In other words, we may not want to detect vulnerabilities in networks that are not of our protected networks. If the sensor was placed on a LAN to watch internal client and server transactions, we may want to substitute an "I" for the "F". The "B" specifies an exact binary match of the search string. In some cases, a signature may be written to detect ASCII characters regardless of case. This option would be specified with the letter "A" instead of "B". The next number specifies how many additional packets Dragon is to log if a successful signature match occurs. For intrusion detection and network forensics, this is very important. However, in most cases we don't need this feature to detect passive vulnerabilities and the number is left as zero. The next number specifies the maximum depth to search a packet. If a packet arrives with a data length smaller than the number specified, the network session is reassembled. The TCP or UDP port is specified next. And finally, the name associated with this signature and the actual signature string are specified. Signature strings may contain HEX characters by escaping them with a "/". For example, "/20" represents a space.

Web servers, FTP servers and many other commercial and open source servers have unique banners that may be searched for. A majority of these also exist on a unique port. However, it may be interesting to search for banner strings on any port to find rogue, illegal, misconfigured or otherwise non-standard network servers. Common examples of this include running unauthorized web servers on ports other than 80, running Internet Relay Chat servers on non-IRC ports and even hacker shell backdoors that also exist on high ports. To modify the above signature to search for any occurrence of the Sendmail 4.1 banner on any port, we would simply change the port from 25 to 0:

T S F B 0 100 0 SENDMAIL:4.1 Sendmail/204.1

Detecting vulnerable servers can be useful if it answers a particular problem. For example, a network administrator may want to run Dragon in order to search for unknown web servers on unknown ports. A signature may be deployed to Dragon for a duration necessary to detect the web server. When to search, what to search for and how long to search for it are up to the local network administrators. Searching for unknown vulnerable servers is a balancing act of finding likely security problems and maintaining a realistic and manageable list of signatures.

Client Based Vulnerabilities

Passive vulnerability detection can be used to detect known network clients that have security vulnerabilities. Once again, we don't detect the vulnerability, we simply detect the existence of the vulnerable client. Many clients identify themselves with unique strings of ASCII and binary data. These strings can be used to identify client activity.

One of the most common Internet clients is the web browser. Netscape and Microsoft have both produced complex web clients that have had long histories of security problems. It is useful for some large organizations to search for usage of older web clients. Here are some Dragon signatures that detect older versions of the Netscape web client:

T D F B 0 100 80 NETSCAPE:2 Mozilla/202.

T D F B 0 100 80 NETSCAPE:3 Mozilla/203.

Each of these signatures searches for a unique string in outbound web sessions from protected networks. If we wanted to, the exact version could have been specified such as this:

T D F B 0 100 80 NETSCAPE:3 Mozilla/204.08

Similarly, email agents can also be sought out. Network organizations may want to collect data on email client usage. Eudora, Netscape and many other "free" email toolkits have a variety of security problems. Searching for old revisions with known security issues is useful for some networks. Many of the clients place unique string information in outbound email messages.

It should be noted that as these events occur, they will occur in groups. In these examples, each email or web request may result in a Dragon signature match. This equates to multiple alerts for each positive match. Dragon's summary and visualization tools can help organize the data into useful information.

Identifying Operating Systems

When accessing a network's overall security posture, it may be useful to search for elusive operating systems with known vulnerabilities. This is a more generalized approach to finding problem systems instead of looking for specific servers. Active scanning identifies a large percentage of hosts that exist, but the technique cannot penetrate most firewalls or detect systems that are offline during the scanning. Passive vulnerability detection can help with this task.

A common scenario involves detecting a small number of users who reconfigure their PCs with an operating system such as LINUX or BSD variant. In most cases, there may be nothing intrinsically wrong with installing a second operating system. In some cases though it is against the network security policy. In all cases, searching for elusive operating systems is still a worthwhile endeavor because they can drastically effect the security posture of a network. As with the old version of Sendmail, it may be possible to search for older operating systems, which may have many security problems.

Two ways developed for Dragon to detect operating systems is to look for distinct server banners that identify the operating system and to look inside web client traffic. In both cases, unique strings of ASCII and binary data may be used to identify specific operating systems.

System Configuration Vulnerabilities

Certain configuration vulnerabilities may also be identified with passive techniques. In this section we will consider searching for network conditions where a configuration setting has resulted in a vulnerability. This is different from searching for servers or clients and merely matching a system or client banner. These configuration vulnerabilities may be detected only when someone is using the service inappropriately. Nonetheless, the event may still be worth searching for. We will consider one example of a Telnet user issuing a command to open up X-Windows security:

T D A B 0 20 23 TEL:XHOST-PLUS xhost/20+/0d

In this example, we assume that issuing the "xhost +" command is a bad thing that should not occur. If Telnet is in use on a given network, then a signature like this may be worth looking for. This type of vulnerability detected passively is highly dependent on the network.

Other examples of configuration problems may include inbound access to sensitive services. Consider the Netscape Enterprise web server. It has a web based management page. Normally, management activity is originated from localhost or from the local network. Typically, external access to such a service is not allowed. Writing a signature to detect unique keywords in outbound network sessions may indicate a configuration problem with servers and possible a firewall, which brings us to our next session.

Firewall Vulnerabilities

Using the term firewall loosely, it is possible to "double check" the firewall rules by using tools to detect network activity that should not occur. For example, lets assume that a simple firewall only allows access to a single web server. It would make sense to try and detect any traffic on the "inside" of the firewall that was going to other web servers. That may indicate a configuration problem at the firewall. It may also make sense to look for any traffic going to ports below 1024 except for the single allowed port 80 web server. Of course this is a simplistic view of a complex problem, but the point is that searching for traffic that shouldn't be there may be a good indication of a firewall configuration error.

Dragon can be configured to search for traffic that should not be there. It can do this a number of ways, but the most direct is with the DESTINATION keyword. Here is an example DESTINATION configuration to detect any other HTTP traffic except that to a known web server.

```
DESTINATION

I 10.100.100.88 6 80

L 10.100.100.0/24 6 80
```

Dragon implements a fall-through access control list. This configuration simply tells Dragon to not alert on any web traffic destined for the web server at 10.100.100.88 and alert on any other web traffic destined for the 10.100.100.0/24 CIDR block.

Active Scanning Pros and Cons

PROS

The biggest advantage of active scanning is its accuracy of the hosts that are scanned. An active scan can connect to as many ports as desired, actually try vulnerability probes and identify the host operating system with TCP fingerprinting to name a few things. The scan is also repeatable, such that trends can be developed to analyze long term security data.

CONS

Active scanning can be very disruptive. Some hosts are fragile and do not handle port scanning and TCP fingerprinting very well. Database servers and mainframes with TCP/IP stacks are notorious for being crippled by tools such as ISS Scanner and NMAP. Log files may also become clogged and alerts are generated in some cases, which cause extra work for network administrators. Certain NAT based firewall and router solutions do not handle internal portscans very well either. Because scanning is disruptive, it has to be coordinated with operations centers. Many times, some sort of legal or administrative permission is required to scan large networks.

Active scanning can take a long time. Doing a full port scan of all possible 64000 UDP and TCP ports for even a Class C address can take a long time. Many times only a small subset of those ports is tested which reduces accuracy, but increases speed. Scans that take a long time to complete also miss "pop-up" servers that periodically connect and disconnect to the network. If the scans always take place during normal working hours, they may miss weekend or evening servers.

And finally, active scans cannot find information about vulnerable network clients such as web browsers.

Passive Monitoring Pros and Cons

PROS

Passive monitoring is not intrusive on network performance or operation. It can also operate 24x7 without human intervention. As an investigative tool, it can be very useful to trace certain network security problems and verify suspected activity.

CONS

Passive network monitoring is completely dependent on the traffic it is watching. Without the traffic, no alerts will occur and vulnerabilities won't be detected. If traffic volumes are high (20-60Mb/s) a dedicated server class sensor may be required. In comparison, an active network probe can be operated efficiently from typical desktop computer equipment. If performance is an issue, a smaller list of signatures may be required to process traffic efficiently.

Conclusions

Passive network vulnerability monitoring is not a replacement for active scanning. It should be viewed as a separate, but complementary technique. Many network security administrators can use a distributed network monitoring tool to troubleshoot a variety of problems. Passive monitoring has several advantages over active scanning, but they are two different techniques. Passive vulnerability monitors will not be able to generate the same amount of raw data that an active security can. However, they are the appropriate tools to search for vulnerable network clients, and elusive servers that are periodic in nature.