Making Effective Use of Your Intrusion Detection System

20th November 2006

Jamie Riden, CISSP (jamesr@europe.com)

New Zealand Honeynet Project (http://www.nz-honeynet.org/)

"... there are people who will try anything to secure their networks, except design them correctly, control the access levels within them, segment their networks, understand their traffic, and monitor things closely." – Marcus J Ranum

More and more network administrators are starting to deploy Intrusion Detection Systems to watch for compromises on their internal networks. However, merely deploying an IDS may not make that much difference to the overall security of the network. The IDS must be tuned to reduce the number of false positives, and to catch as many genuine attacks as possible. Then an analyst must be available to monitor the alert stream and appropriate action must be taken to deal with alerts. Without effective response, the IDS will be of little use. In this article we talk about optimal IDS placement in your organisation, how to correlate alerts with other data sources, how to tune the IDS rule sets and how to respond to a variety of alert types.

Introduction

The attacker has a lot of advantages on the Internet; he or she may be hard to trace and may have a great deal of time and equipment to spend mapping out a network's weak points before they launch an attack. Worms and viruses may be able to exploit weaknesses very rapidly before a human can carry out a proper incident response. However, the defender has two big advantages. Firstly, the administrator can achieve excellent visibility of what is happening on their network, via logs, audit trails and other monitoring systems. The second advantage is domain knowledge - the defender should have a good idea of what traffic can be expected from the various computers on the network, which makes it easier to detect attacks.

Visibility and Logging

Visibility is the best word I can think of for having plentiful, accurate data about what is currently happening on your network. The attacker doesn't know your network, and doesn't have access to the wide variety of logging data, audit trails and network statistics that you do. The administrator should log as much data as is practical; even if you don't make a habit of reading it day-to-day, it will hopefully be available if you need to investigate an incident such as a compromise. It may also provide evidence to be used in disciplinary proceedings. If you have data which goes back for a significant period, you may be able to check current anomalies against historical trends to see if they are something you should really be concerned about. However, the best idea is to get to know your logs. Dip in from time to time and see what's happening. Are you surprised? Is it different from last week?

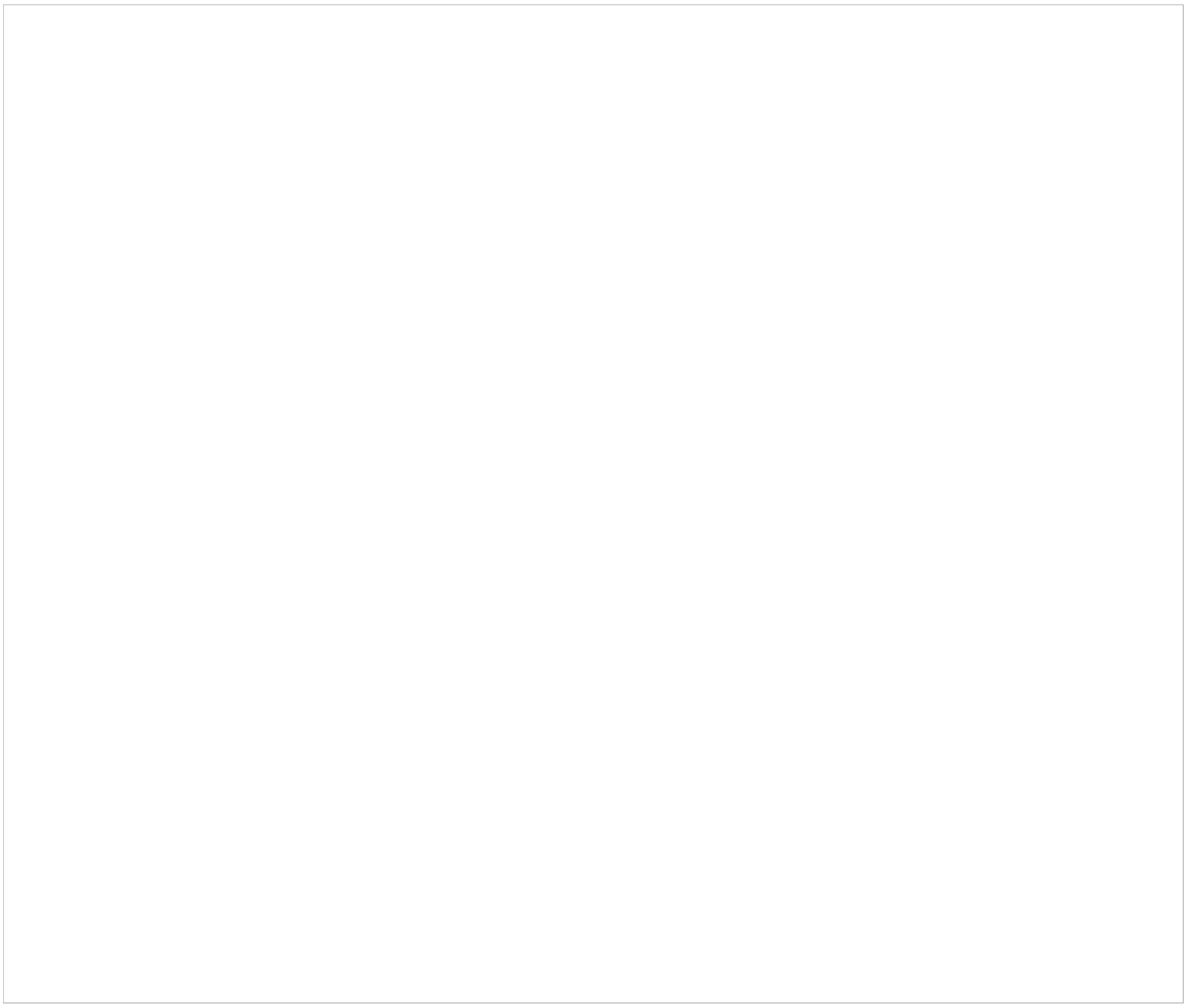

Figure 1: Traffic summarisation from [ntop]

One point to note about audit trails is that they only tell you the logged in user; hopefully you have a strong password policy, or are even using two-factor authenticaton, and have a strong belief that the username corresponds to the person behind the keyboard. If it is relatively easy to guess or acquire someone else's password, you cannot be sure of this correlation any more and your audit trail becomes of very little use when reconstructing events and almost none in disciplinary proceedings.

I recommend you send all logging information to a central host, as well as keeping a few days around on individual servers. You can use something like syslog-ng for this, though some devices such as BlueCoat web caches will need to use FTP upload to transfer logs. It doesn't hurt to use a program such as tripwire to make sure these files do not change, and to back up the data to tape as well.

```
# Options for start/restart the daemons
#   For remote UDP logging use SYSLOGD="-r"
#
SYSLOGD="-r"
```

Figure 2: Your /etc/init.d/sysklogd script probably contains something like this. Make sure you start syslogd with a -r argument to enable it receive remote logging packets on port 514/udp.

```
auth.*                     @loghost.example.com
```

Figure 3: Configuring authentication messages to go to a remote log host, in /etc/syslog.conf

Domain Knowledge

All programmers will liberally sprinkle assert statements in their programs. These are intended to catch 'impossible' conditions. If you know the output of a particular function should always be positive, then you write `assert(result>=0);` in the function. After all, even if you write the function perfectly, someone may introduce a bug in the future, or call it with invalid parameters.

```
alert tcp $EXTERNAL_NET any -> $INTERNAL_NET 23 (msg:"TELNET attempt inbound";
flow:to_server,established; classtype:misc-activity; sid:xxx; rev:1;)
```

Figure 4: telnet is banned by company policy - snort rule to log the traffic.

```
-A LOGDROP -m limit -j LOG --log-prefix "filter: "
-A LOGDROP -p tcp -j REJECT --reject-with tcp-reset
-A LOGDROP -j REJECT --reject-with icmp-port-unreachable
-A LOGDROP -j DROP
...
-A INPUT -p tcp -m tcp --dport 23 -j LOGDROP
```

Figure 5: telnet is banned by company policy - iptables rule to log and drop the traffic

In the same way a network administrator has a pretty good idea about things that should never happen. Your web servers should never join an IRC channel, nor should they usually initiate connections to the Internet. Your traffic volume should be highest in the morning and afternoon and be relatively quiet outside working hours. No machine should send try to initiate more than five to ten TCP connections per second for any sustained period. Some of these will vary by site, but the overall priniciple holds. So it makes sense to be alerted whenever anomalous events occur; you can always disable that alert if it should prove to be unhelpful. If you don't set the alert you may never find out about malicious activity.

Some of the things you will find out will not be malicious intruders. They may be people downloading episodes of 'Friends' using BitTorrent, synchronising a Debian mirror, or just running nessus against your servers "to see how secure they are". Then again, you may have a compromised machine and the sooner you find out about it, the less damage the attacker can do.

Where do you put your IDS?

There are several different schools of thought - one says, put your IDS outside your firewall, then you can see all the attacks and probes that are launched at you. Another says that this is too much for a normal analyst to cope with and why bother with attacks that are stopped by the firewall? Put it on the internal network, that's what we really need to keep an eye on.

I would probably have to go with option two. Put an IDS outside the firewall if you want, but don't try to read it day-to-day; some of the data will be duplicated by the firewall logs anyway. You are keeping your firewall logs for a month or so aren't you? The place where you really need IDS is inside your network, on the DMZ and the internal network. If you have a compromise there, you need to know as soon as possible, so pay close attention to the alerts.

You will find it worthwhile to look for malicious traffic exiting your network as well as entering it - it should be much rarer and finding signs of attacks being launched from within indicates you have a real, urgent problem.

Figure 6: You may wish to put your IDS on a SPAN port on your core router

Common behaviours of malware

Textbooks will make a distinction between worms, viruses, trojan horses and logic bombs. While this is a useful distinction, an administrator is often interested in other things. Firstly, worms such as Blaster and Sasser spread autonomously and in general do not damage data, nor report home via a central channel. Other worms such as Witty will corrupt data, and some other worms may try to leak documents to a third party. Bots such as Rbot, Sdbot, Gaobot can act like worms for some of the time; in particular their scanning activity will can like a worm while it is in progress; however they are controlled, often via an IRC channel, so blocking this channel is probably the first move in remediation. If a machine is compromised via a manual attack, the attacker may give themselves away for scanning large numbers of hosts for the same vulnerability they exploited to own your machine. Other bots may use Google to search for infected hosts.

```
-----------------------------------------------------------------------------
#(4 - 1329104) [2005-03-25 03:39:49.297] [snort/2001372]
BLEEDING-EDGE IRC Trojan Reporting (Scan)
IPv4: yyy.yyy.231.32 -> zzz.zzz.163.59
      hlen=5 TOS=0 dlen=168 ID=18140 flags=0 offset=0 TTL=127 chksum=56572
TCP:  port=3023 -> dport: 8000  flags=***AP*** seq=1483308911
      ack=501861482 off=5 res=0 win=64331 urp=0 chksum=51363

Payload:  length = 128
000 : 50 52 49 56 4D 53 47 20 23 61 73 74 72 6F 20 3A   PRIVMSG #astro :
010 : 5B 53 43 41 4E 5D 3A 20 52 61 6E 64 6F 6D 20 50   [SCAN]: Random P
020 : 6F 72 74 20 53 63 61 6E 20 73 74 61 72 74 65 64   ort Scan started
030 : 20 6F 6E 20 yy yy yy 2E yy yy yy 2E 78 2E 78 3A    on yyy.yyy.x.x:
040 : 34 34 35 20 77 69 74 68 20 61 20 64 65 6C 61 79   445 with a delay
050 : 20 6F 66 20 35 20 73 65 63 6F 6E 64 73 20 66 6F    of 5 seconds fo
060 : 72 20 30 20 6D 69 6E 75 74 65 73 20 75 73 69 6E   r 0 minutes usin
070 : 67 20 32 30 30 20 74 68 72 65 61 64 73 2E 0D 0A   g 200 threads...
```

Figure 7: A captured packet showing a bot reporting to an IRC server -
this alert was sent by BASE console for snort.

You cannot in general rely on antivirus software to detect all your intrusions. "I always say, 'Do not rely on antivirus software'. The problem is we have to see a virus before we can detect it. If antivirus is in place, it may not detect [a virus] because it may be very targeted for you."[1] says

Vanja Svajcer of Sophos. Graham Ingram of AusCERT reinforces this: "At the point we see it as a CERT, which is very early on -- the most popular brands of antivirus on the market... have an 80 percent miss rate. That is not a detection rate that is a miss rate."[2] In these circumstances, you need to observe and investigate anomalous behaviour on your network. If you are new to forensic analysis, I strongly recommend Forensic Discovery by Dan Farmer and Wietse Venema.

For example, a nepenthes honeypot that the New Zealand Honeynet Project recently set up has captured 194 different malware binaries between 12th and 29th October. Roughly 70% of these were identified as malware by a particular virus scanner on the 29th October, with the other 30% not being identified. Since all were dropped by exploiting particular Windows vulnerabilities, we can safely assume that the 30% which are not identified are also malware.

Common vectors for attack

People inside your organisation may receive malware via email, or may be prompted to download it from a website. Alternatively they may get files, or links to files through instant messaging, or peer to peer file sharing programs. Some malware can act when the user merely views a web page. Post-compromise, the malware may try to spread via similar mechanisms to the one it used to infect the host in the first place; via email, or port-scanning looking for machines potentially vulnerable to the same exploit. Often the attacker is just looking to compromise as many machines as possible and will be using yours as a staging post to expand their empire.

```
Jun 26 22:31:04 victim sshd[15384]: Failed password for root from ::ffff:w.x.y.z
port 30937 ssh2
Jun 26 22:31:06 victim sshd[15386]: Illegal user network from ::ffff:w.x.y.z
Jun 26 22:31:06 victim sshd[15386]: error: Could not get shadow information for
NOUSER
Jun 26 22:31:06 victim sshd[15386]: Failed password for illegal user network
from :: ffff:w.x.y.z port 30951 ssh2
Jun 26 22:31:08 victim sshd[15388]: Illegal user word from ::ffff:w.x.y.z
Jun 26 22:31:08 victim sshd[15388]: error: Could not get shadow information for
NOUSER
Jun 26 22:31:08 victim sshd[15388]: Failed password for illegal user word from
:: ffff:w.x.y.z port 30963 ssh2
Jun 26 22:31:10 victim sshd[15390]: Failed password for root from ::ffff:w.x.y.z
port 30980 ssh2
Jun 26 22:31:11 victim sshd[15392]: Failed password for root from ::ffff:w.x.y.z
port 30992 ssh2
Jun 26 22:31:13 victim sshd[15394]: Failed password for root from ::ffff:w.x.y.z
port 31007 ssh2
Jun 26 22:31:15 victim sshd[15396]: Failed password for root from ::ffff:w.x.y.z
port 31021 ssh2
Jun 26 22:31:17 victim sshd[15398]: Failed password for root from ::ffff:w.x.y.z
port 31031 ssh2
Jun 26 22:31:19 victim sshd[15400]: Failed password for root from ::ffff:w.x.y.z
port 31049 ssh2
Jun 26 22:31:20 victim sshd[15403]: Failed password for root from ::ffff:w.x.y.z
port 31062 ssh2
Jun 26 22:31:22 victim sshd[15405]: Failed password for root from ::ffff:w.x.y.z
port 31073 ssh2
```

Figure 8: Entries in /var/log/syslog from an attack againt ssh

While an incoming attack may be very difficult to detect in among all the Internet noise, an outgoing attack should be very easy to spot, as it is (or should be) very different from your normal network traffic. Look for excessive volumes of traffic of certain types or from individual hosts. See Extrusion Detection: Security Monitoring for Internal Intrusions by Richard Bejtlich for more on detecting internal attacks.

The logs in Figure 8 were recovered from a system that was eventually compromised when the attacker tried the username/password combination of upload/upload. This didn't set off any alarm bells, but the attacker spent the weekend scanning for systems with open port 22 (sshd), which generated hundreds of thousands of portscan alerts on the IDS console.

Correlation

IDS alerts can be ambiguous, even when you have packet captures to correspond to these alerts. It's always useful to check on what other alerts have been triggered from the same source, and what other sources have triggered the same alert. The former might reveal a studied attempt to probe weaknesses in your system, the latter will tell you if you have a couple of machines with similar behaviour patters, - often bots, worms or file-sharers - active on your network.

In the following example, we're using argus to maintain a list of connections, and the reporting tool 'ra' to list them out from the current log file. The -z option determines how the TCP options are output. The first connection was a completed request for a web page, outbound. The second is an inbound request for a webpage, and the third represents an ARP query and reply.

```
# ra -z -r /var/log/argus/argus.log
10-29-06 13:40:09.322643           tcp                  192.168.0.8.3402
->                  216.73.87.52.www        4        3          825          690
sSEfF
10-29-06 13:36:38.041997           tcp                219.33.24.157.2916
->                   192.168.0.8.www        5        4          468          224
sSEfF
10-29-06 13:43:15.733802           arp                  192.168.0.1
is-at               0:14:6c:67:cc:9a          1        1          42            60
CON
```

Figure 9: Using argus to view lists of connection records.

You will want to have easy access to logs from your DNS, DHCP, web, ftp and mail servers (both in- and outbound), as well as your firewall and if possible flow records from a program such as argus. These will all help you to reconstruct an incident, or to satisfy yourself all is well. This is where having all the logs on a single host will come in useful. In the previous example, we saw portscanning activity from a single host on port 22/tcp, and the obvious guess was that this had been compromised via the same exploit. Looking at the sshd entries in the syslog revealed a password guessing attack which was eventually successful.

Honeypots

If you can install a few honeypots - systems with no legitimate purpose and good monitoring capabilities - it may make the job of analysing traffic very much easier. If a system that you're examining has touched your honeypot, it becomes very much more suspicious. Tools such as nepenthes will collect pieces of malware that certain worms and attackers are trying to deliver. By placing such a system inside your firewall, you may get early warning of some classes of worms and bots penetrating your network. For example, if you are using 10.0.0.0/8 internally, you can use a single machine running nepenthes to respond to all addresses within perhaps a /20 or more. You may also capture malware binaries which can come in useful if your antivirus vendor doesn't include that particular threat in their signatures at the time. In this case, you need to submit the unidentified malware to them as soon as possible.

Figure 10: Using some of your internal address space as a honeynet.

Tuning IDS rule sets

My favoured approach is to turn everything on and then disable alerts which are of little or no use. While this has a very large initial investment, it will help in two important aspects: One is that if you only turn on alerts for things you are looking for, you may not pick up events that are unexpected, but nevertheless unwelcome. Perhaps someone has ignored change control, installed an application on a server and then forgotten about it or has left the company. Second, it will help acquaint you with the normal types of traffic that can be found on your network. When you've investigated an alert and are fairly sure that all the events are false positives, you can disable it. If there are few false positives, leave it there in case there is a real event further down the track.

The other way to reduce false positives is to reduce or block that type of traffic flow completely. For example, do your users have a business need to be using Internet Relay Chat? If not, block it and save yourself looking through alerts for botnet Command & Control channels - the IRC-based ones simply won't work and you can save yourself some time. Equally, don't worry about alerting on DNS traffic to other than your internal DNS servers if you can stop it all at your perimeter. Keep the rules in place to make sure someone hasn't broken your firewall ruleset though.

Response

Remember that preventing an incident can be much easier than responding to one. If you see people trying to guess passwords for SSH, why not move the service to another port, or block it from the majority of the Internet? Even if you're sure that none of the passwords are guessable now, things change. Maybe your password checker (for example, john the ripper) will be trying to email you a list of easy to compromise passwords, but they may end up in the spam filter without reaching you. But, however much prevention you do, you will still have incidents.

The first step, which we have already discussed is to detect incidents. Then pick your best hypothesis - worm, bot infection, denial-of-service, cracker - and contain the affected machines. If it's a worm, block the propagation mechanism; if it's a bot, block the control channel and the vectors used to spread. Do what you can about a DoS and alert your upstream provider(s).

For example, you can ask snort to tear down connections based on a signature match, via the flexible response feature. The following rule attempts to disrupt the transfer of Sasser via ftp over port 9996; how effective it will be depends on a number of factors, but may be worth trying in dire straits.

```
alert tcp any any -> any 9996 (msg: "BLEEDING-EDGE VIRUS Sasser Transfer
_up.exe"; flow: established,to_server; content:"|5F75702E657865|"; depth: 250;
classtype: misc-activity;
reference:url,vil.mcafeesecurity.com/vil/content/Print125009.htm; resp: rst_all;
sid:2000047; rev:4; )
```

Figure 11: snort will send TCP RST packets to both ends of the connection when this signature is triggered.

After the incident has been contained to a greater or lesser extent, you need to develop a remediation procedure. If you have many machines, write down a series of mechanical steps to fix each machine and distribute it to the technical consultants. If the incident is small, it will still come in useful when you write up your report. Usually the real solution is to re-install the machine because you don't know what binaries have been tampered with. You may have to disable the Ethernet outlets affected machines are plugged in to - can you rapidly match an Ethernet port to a particular machine?

If you don't have the necessary tools, it is best to develop them before an incident occurs. Doubtless you will think of enhancements you could add during the course of the clean-up, but it's considerably better than having no tools to work with.

Conclusion

Get to know your network: the typical activity, or baseline. What things should you expect in logs? What things shouldn't be there? Make sure you have the tools and procedures in place ready for an incident. Draw on as many data sources as possible when analysing the course of events. If one of your security measures fails, what will happen? If an attacker breaches the network perimeter, will he have unrestricted access to the whole internal network? Will you notice an attack from one internal machine to another?

Properly implemented, IDS can be a great asset when detecting intrusions as well as providing information on how your containment and eradication is progressing. It works best when combined with the two main advantages of the defender: domain knowledge and visibility. As the defender, you are in a position to know the topology and make-up of your network and you have ready access to exactly what is going on. Use both of these to your best advantage and you will find that incident detection and response becomes that much easier.