Network Security

Professor Gaurav S. Kasbekar

Department of Electrical Engineering

Indian Institute of Technology, Bombay

Week - 10

Lecture - 56

Firewalls and Intrusion Detection Systems: Part 6

Hello, recall that in the previous lecture, we introduced intrusion detection systems; in this lecture, we will look at different types of IDSs. So, IDSs can be classified in various ways. One classification is based on the detection method, that is, how the IDSs detect malicious traffic. Another classification is based on where detection takes place. That is, does it take place in the network or on the host?

The classification based on the detection method is as follows: The IDSs are divided into two types. One is signature-based systems, and another is anomaly-based systems. And the classification based on where detection takes place classifies IDSs into host-based systems and network-based systems. So, let us discuss these four types of IDSs.

Signature-based systems, anomaly-based systems, host-based systems, and network-based systems. We will start with signature-based IDSs. A signature-based IDS maintains an extensive database of attack signatures. What is an attack signature? Each signature is a set of rules used to match a packet or a series of packets with a known intrusion activity.

- E.g.:
  - Snort is a public-domain, open source IDS
  - An example of a Snort signature is as follows
  - alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg: "ICMP PING NMAP"; dsize: 0; itype: 8;)
  - This signature is matched by any ICMP packet that enters the organization's network ($HOME_NET) from external network ($EXTERNAL_NET), is of type 8 (ICMP ping packet) and has an empty payload (dsize = 0)
  - Since nmap generates ping packets with these specific characteristics, this signature is designed to detect nmap ping sweeps

Here's an example of an attack signature. Snort is a public domain, open-source IDS. You can search online for more information about Snort. Here's an example of a Snort signature. alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg: "ICMP PING NMAP"; dsize: 0; itype: 8).

So, what does this signature mean? This signature is matched by any ICMP packet that enters the organization's network that is indicated by this $HOME_NET from the external network that is indicated by this $EXTERNAL_NET and is of type 8. Type 8 ICMP packets are ping packets and has an empty payload, that is, d size equals 0. So, what is so special about all these characteristics? Nmap which we discussed in the previous lecture generates ping packets with these specific characteristics. Hence, this signature is designed to detect nmap ping sweeps.

So, if an attacker is trying to scan the network using nmap, then nmap will send a sequence of packets with all these characteristics. And using this signature, an IDS can find out that it's probably an nmap packet. So, this is an example of a signature which can be used to detect network scanning. In general, a signature may be a list of characteristics about a single packet. For example, source and destination port numbers, protocol type, and a specific string of bits in the packet payload.

Or it may relate to a series of packets. So, for example, there may be a series of some five packets, each of which has part of a virus payload. So, a signature may pertain to one packet or to a series of packets. Signatures are normally created by skilled network security engineers who research known attacks. These engineers study attacks that are launched by malicious users on networks, and they examine the kinds of packets that are sent into the network by these malicious users, and by examining such packets, they create signatures which can be used in the future by IDSs to match with the traffic that they observe and detect attacks.

A signature-based IDS operates as follows. It sniffs each packet passing by it and compares it with each signature in its database. There may be tens of thousands of signatures in the database. If a packet or a series of packets matches a signature in the database, the IDS generates an alert. Here are some examples of detection by a signature-based IDS.

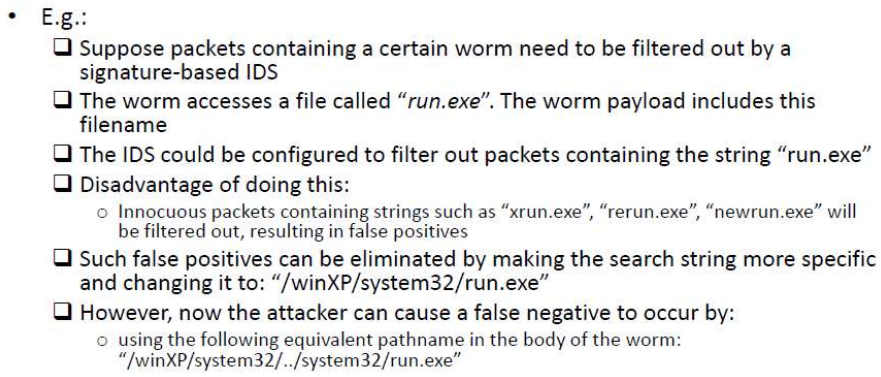
Suppose there is an attempt to establish a TCP connection from an external host to an internal host at an unused destination port. So, what does this indicate? It may indicate an attempt to discover which services are open. So, some attacker is possibly sending packets to different port numbers, and when it receives a response, it knows that there is an application process at that port number. So, this may indicate an attempt to find out which services are open, that is there are processes at which port numbers.

So, this is a kind of scanning which can be detected using signature based ideas. If a specific byte sequence is present in the application payload of an incoming packet, then it may

indicate an attempt to inject specific malware, for example, virus or worm. So, this byte sequence may be a part of the virus or worm, so that may indicate that there is an attempt to inject such malware. So, by matching the specific white sequence with one of the signatures, the IDS is able to detect the presence of a virus or a worm. An attack that is undetected by a signature-based IDS is called a false negative, and an IDS is set to generate a false positive if it raises an alarm even when there is no ongoing attack.

Both false positives and false negatives should be minimized. For example, the false positive rate should be less than 1% or 5% or so. Similarly, the false negative rate should be less than 1%, and so on. So, we typically have targets about how much we can tolerate false positives and false negatives. But there is a trade-off in doing this.

If we reduce false positives, then typically that is at the expense of increased false negatives, and vice versa. So, here's an example. Suppose packets containing certain worm need to be filtered out by a signature-based IDS. So, the worm accesses a file called "run.exe", and the worm payload includes this file name because the worm must access that file. So, the file name is included in the worm's payload.



The IDS could be configured to filter out packets containing the string "run.exe". So, this will detect the worm, and it detects the worm by looking at the presence of this string run.exe. But there is a disadvantage of doing this. Innocuous packets which contain strings, such as "xrun.exe", "rerun.exe", and "newrun.exe", these will also be filtered out because this string, "run.exe", is present in all of these innocuous packets. So, these packets will also be filtered out, and that will result in false positives.

Actually the packets are not malicious, but the ideas flags them as malicious packets. Hence, there is a false positive. So, such false positives can be eliminated by making the search string more specific and changing it to the full path name, that is

"/winXP/system32/run.exe". But now the attacker can cause a false negative to occur by using the following equivalent path name in the body of the worm. That is /winxp/system32/... This '..' refers to the parent directory and then again system32 and then /run.exe. So, this path name is refers to the same file as this path name refers. But the search string that is used in the signature database that matches only this string. It does not match this string. So, hence, this payload will not be detected, and that will cause a false negative to occur.

So, via this example, we see that there is a trade-off between false positives and false negatives. So, if you make the string for comparison very general, such as "run.exe", then there'll be a lot of false positives because strings such as "xrun.exe", "rerun.exe", and so on will also be matched. But if we make the search string very specific, then some strings, such as this one, will not be matched by the IDS, so hence there'll be false negatives. So, this shows that there is a trade-off between false positives and false negatives. Signature-based IDSs have a limitation.

They require prior knowledge of the attack to generate a signature. So, some security researcher must analyze an attack and analyze the kinds of packets that are sent during the attack and then create the signature. Only then, in the future, can that attack be detected. So, for this reason, signature-based IDSs are unable to detect attacks that have not yet been recorded. Since each packet needs to be compared with an extensive collection of signatures, the IDS can become overwhelmed with processing, and hence it can fail to detect some malicious packets in a timely manner.

So, part of this problem is alleviated by having multiple IDS sensors. So, each sensor only sees a part of the traffic that flows into the organization's network. But despite this, since each packet needs to be compared with an extensive collection of signatures, the IDS can become overwhelmed with processing, and hence, it can fail to detect some malicious packets in a timely manner. So, this is another limitation of signature-based IDSs. So, we have discussed signature-based IDSs.

Another kind of IDS is anomaly-based IDS. It does not operate by matching the packet's payload with some pre-stored signatures. Instead, it analyzes the statistics of the traffic that flows through the network. It stores the statistics of the traffic observed during normal operation. And it looks for packet streams that are statistically unusual.

Some examples of statistically unusual traffic streams are as follows. One example is a large percentage of ICMP packets. So, if the percentage of ICMP packets is usually below

a certain threshold and then suddenly the percentage of ICMP packets is above that threshold, then that may indicate a statistically unusual packet stream. What is the advantage of anomaly-based IDSs? They don't allow previous knowledge about existing attacks; so, hence, they can detect new attacks which cannot be detected by signature-based IDSs.

They can potentially detect new and undocumented attacks. Some examples of anomalous packet streams are as follows. Suppose there is a tenfold increase over the norm in the number of accesses to a specific file; then that may indicate that a denial-of-service attack is happening. So, this is an example of an anomalous packet stream. If the login frequency to a particular account is unusually high, then that may indicate that someone is trying to break into that particular account.

So, that attacker is trying out different passwords and trying to log into that account. If the number of distinct source IP addresses of packets arriving at the organization's network from outside is very high, then that may indicate a distributed denial-of-service attack. So, we discussed earlier how a distributed denial-of-service attack is performed. An attacker initially compromises many hosts and enrolls them in what is known as a botnet, and packets are sent from all these compromised hosts to the victim. So, in a distributed denial-of-service attack, the number of source IP addresses of packets arriving at the organization's network from outside will be very high, and this can be used to detect the attack.

If the ratio of the number of TCP SYN packets to the number of TCP FIN packets in a time interval is much greater than one, then that may indicate a SYN flooding attack. So, recall that every TCP connection starts with a three-way handshake, which includes TCP SYN packets, and the connection ends with TCP FIN packets. So, if the ratio of the number of SYN packets to the number of FIN packets is much greater than one, then that indicates that an attacker is trying to send a lot of SYN packets to create a lot of half-open connections. So, this indicates a SYN flooding attack. So, recall that earlier we discussed how SYN flooding attacks can be defended against using the mechanism of cookies.

This is another way in which we can defend against SYN flooding attacks by using anomaly-based ideas. But anomaly-based IDSs also have a disadvantage. It is challenging to distinguish between normal traffic and statistically unusual traffic. Some challenges in distinguishing between these kinds of traffic; some challenges are as follows. The IDS will have to learn over time what constitutes normal activity, and this can be challenging.

The definition of what is normal may vary as a function of the time of day or day of the week. For example, the kind of traffic that normally flows in a network during mornings may be different from the kind of traffic that flows at night. Similarly, the traffic on weekdays and weekends may be different. Hence, the IDS will have to learn what normal activity is, and this normal activity varies with the time of day or day of the week. What is normal may also vary from one host to another since different users have different traffic patterns.

So, traffic that is normal for one user may not be normal for another user, and vice versa. For these reasons, to date, most IDS deployments are signature-based, but some IDSs also include anomaly-based features. We have discussed the classification of IDSs based on detection method. The classification was signature-based IDS and anomaly-based IDS. Now we discuss another classification based on where the IDS resides—whether it resides in a host or in a network.

An IDS that captures information about packets flowing through the network is referred to as a network-based IDS. For performance reasons, it is common to have standalone appliances that perform network-based intrusion detection. These standalone appliances only run the IDS and hence are not vulnerable to malware attacks. If we had an IDS which ran in software on a PC, in that case its performance would be low. In contrast, if we have a standalone appliance that performs network-based intrusion detection, then that typically performs faster than an IDS which is running in software on a PC.

So, a network-based IDS is typically deployed at multiple points in a large organization. In the previous lecture, we discussed an example network architecture in which there were three IDSs. A host-based IDS is typically implemented in software and runs as an application on a host. So, it monitors the internal behavior of the host. For example, it analyzes the sequence of system calls that are made, and the files that are accessed, and so on and so forth.

A host-based IDS makes use of the following to identify events related to an intrusion. The IDS looks at operating system logs, application logs, and so on. So, operating system logs, as the name suggests, they have information about the operating system, that is, log information about operating systems, and application logs have log information about applications. The operating system logs keep track of when users log in, the number of unsuccessful login attempts, commands executed, network connections made, and so on

and so forth. And application logs keep track of the events that are logged by an application during its execution.

For example, which files it opened, which system calls it made, and when. So, such information is analyzed by host-based IDS. Tracking modified files can play a key role in host-based intrusion detection. For example, a change in the contents of core system libraries should cause suspicion because these are rarely modified, if ever. So, by detecting file modifications, a host-based IDS can infer that malicious activity is occurring.

File system integrity checkers compute a cryptographic hash on the contents of each file, and they compare the computed hash of a file to its stored hash. So, when a system is initialized the integrity checkers they compute a cryptographic hash of the contents of each file and store that hash. Later on, from time to time, they compare the cryptographic hash of the file with the pre-stored hash, and if there is a discrepancy, it indicates that the file has been modified. And depending on the kind of file, that may indicate some malicious activity. For example, there should rarely be any change in the contents of core system libraries.

So, if there is a mismatch in the hash, then that indicates that there is possibly an ongoing malicious attack. So, this summarizes the operation of a host-based IDS. So, in summary, we discussed different types of intrusion detection systems. We discussed signature-based IDSs and anomaly-based IDSs. So, based on the detection method, IDSs are classified into signature-based IDSs and anomaly-based IDSs.

And we discussed network IDSs and host-based IDSs. So, depending on where the IDS is deployed, they are classified into network-based IDSs and host-based IDSs. We'll continue our discussion of IDSs in the next lecture. Thank you.