A High Level Overview of

INTRUSION DETECTION SYSTEMS

Joseph Tsai

Joseph Tsai
Overview of Intrusion Detection Systems

Table of Contents

Joseph Tsai
Overview of Intrusion Detection Systems

Purpose of This Document

This document is *not* a comprehensive guide or explanation to Intrusion Detection Systems (IDS's) by any means. It is simply a reflection of my own studies into the topic, with corresponding pictures obtained from various lecture slides produced by Network Systems and Security professor, Professor Geetha Thamilarasu.

The only association that this overview has in relation to Professor Thamilarasu is the usage of the pictures in the sections below. Should any ideas be found inaccurate, this is a reflection of my understanding, and my understanding alone.

Please do let me know in the case that my understanding is incorrect. I would be happy to fix it, as needed.

Intrusion Detection Systems (IDS's)

IDS's. Systems that detect intrusions. At first glance, these systems are pretty self-explanatory, no?

Upon a deeper dive, one will find that there are multiple intricacies to IDS's. In addition to providing a high-level overview of IDS's, this document will also explore some of those intricacies.

Defining "Intrusion"

There could be debate on this definition, but I define intrusion as, "Any unexpected or malicious action taken against an asset." Here's why:

- **"Unexpected"** indicates actions where non-malicious users do something that may seem malicious, but were truly unintentional.
    - **Example:** System administrator accidentally turns off a firewall to an application.

- **"Malicious"** calls out the "bad actors" who intentionally want to do harm against our systems.
    - **Example:** Gaining access to a corporate network and sniffing network traffic.

- **"Asset"** relates to the systems and applications we (or our customers) care about.
    - **Example:** A database which stores sensitive customer information.

Intrusion = Past the Firewall

Usually, our first line of defense are our firewalls. These are our, "keep the bad guys out" mechanisms, which, don't get me wrong, are quite helpful. However, the best IDS's must operate under the "assumption of breach", where they *assume* that malicious attackers have already entered in the network. As also seen in the prior definition, we also care about internal users who have accidentally performed malicious looking actions.

The point is, there are limitations to firewalls that we must accept. No offense to firewalls intended- this is simply a fact that must be considered when building our systems with “defense in depth” at the forefront of our minds.

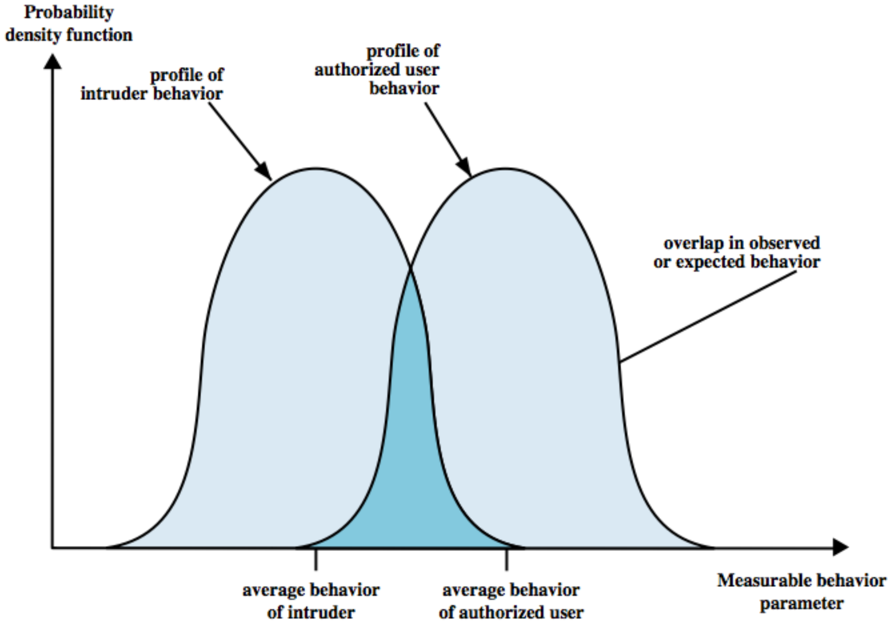
We can think of firewalls as our first line of defense, and IDS’s as our second line of defense. They work together to ultimately provide a more secure environment for all users of our systems.

IDS Principles and the Core IDS challenge

The core assumption of many IDS’s is that intruder behavior is different from legitimate users. Now, is this true?

Consider the example provided previously where an application firewall is disabled. Do we know if it was a legitimate user or malicious user? What if an attacker was able to gain access to a legitimate user’s account? What if the attacker was able to perform the attack from the lobby of our building, thus providing them an IP address that we inherently trust?

We can think of this problem as two overlapping bell curves:


Probability density function
profile of intruder behavior
profile of authorized user behavior
overlap in observed or expected behavior
average behavior of intruder
average behavior of authorized user
Measurable behavior parameter

The point is, there can be an *overlap* between expected behavior of a legitimate user, and the expected behavior of a malicious user.

The key challenge that we face when using IDS’ is to determine this delineation between our legitimate users and the malicious users.

Different types of IDS alerts

With this overlap, we come to the understanding that not all alerts are going to lead to legitimate incidents. That is, we may have an alert be created when a legitimate user is doing something they may be expected to.

On the other hand, we come to the realization that we may not capture all malicious actions, if we don't fire off enough alerts.

It is this balance that we seek, and relates to the core challenge to IDS systems which we discussed in the previous section.

The different scenarios that an IDS can present are seen below:

- **False Positive:** Alert created, but not a true incident
- **False Negative:** No alert created, but it was a true incident (attack undetected)
- **True Positive:** Alert created for a true incident (attack detected)
- **True Negatives:** No alert created, with no associated incident

Goals of IDS's

With these definitions, we can say that the goals of IDS's are to do the following:

1. Maximize true positives (detecting true attacks)
2. Minimize false positives (too many alarms for non-attacks) and false negatives (not creating enough alarms for true attacks)
3. Minimize the time spent verifying attacks and looking for them (AKA – detect true attacks ASAP)

The Base Rate Fallacy

With these definitions in mind, it makes logical sense that we want to strike a healthy balance of detecting true attacks, while not creating alarms for all activity on the network. Here's why:

- If we create too many alarms, then alerts are going to be ignored.
- If we don't create enough alerts, then attacks are going to remain undetected.

This brings us to the concept of the base rate fallacy. The scenario is this: An IDS vendor walks up to you and says, "Greetings, security information professional! I have a great system that will detect all kinds of intrusion attempts known to mankind! In fact, my system is so great, that it detects 99% of attacks on your network. Proven by security information professionals!"

Let's say for purposes of explanation that you end up purchasing this system. And you know what- the IDS salesperson was absolutely right. You *do* capture 99% of attacks on your network. But you know what else? You're getting 1000 alerts an hour, every single day! Why is this the case?

You see, to have such a high detection rate for attacks, this vendor has used extremely sensitive rules in the IDS. The extreme case would be for this vendor to say, “I capture ALL attacks on your network. I just alert on every packet that the IDS picks up!”

Let’s explore this a bit further, using some numbers to help illustrate the point.

Again, let’s use the example of the IDS that can detect 99% of attacks, or 99% of malicious packets.

- The true positive rate in this case is 0.99 (alert fires for a true attack)
- That makes the false positive rate 0.01 (alerts fires for a non-true attack)

Now, let’s say you have 10,000,000 packets on your network, and that 1/100,000 packets on your network is malicious.

Happy with your new purchase, you start up the IDS that the vendor has sold you. After a couple seconds, you get your first alert of a malicious packet on your network. Great! Let’s take a look at the likelihood that the given packet is *actually malicious*.

- 10,000,000 packets total. 1/100,000 packets are malicious, so that means we can assume out of the 10,000,000, that 100 packets are malicious (10,000,000 * 1/100,000).
- Remember, the alarm raises for 99% of these malicious packets. That gives us 99/100 of those malicious packets having an associated alarm. Unfortunately, that also means that 1/100 of our packets doesn’t have an alarm raised.
- What about our legitimate packets, though? 10,000,000 packets minus the 100 malicious packets gives us 9,999,900 legitimate packets.
- Remember that false positive rate we calculated? Those are the scenarios where an alert is raised, but not for a malicious packet. This was calculated to be 0.01. That means for 1% of the legitimate traffic, that an alert gets raised. 0.01 * 99,999,900 gives us 99,999 false alarms
- We can sum all the alerts together to get the total number of alerts that would be raised out of this 10,00,000 packets. 99 alerts for malicious packets plus the 99,999 false alarms gives us a total of 100,098 alerts that are raised per 10,000,000 packets.
- Back to the original question. We get told that a packet is malicious. What is the probability that it is actually malicious?
- Well, we just take the 99 true positive alerts divided by the total number of alerts (100,098), which gives us a 0.000989, or 0.0989% chance that the alerted packet is actually malicious.

I say this jokingly of course, but have fun triaging those alerts!

Here is a table that visualizes the calculations performed above:

Joseph Tsai
Overview of Intrusion Detection Systems

	Malicious Packets	Legitimate Packets	Total
Alarm Raised	99	99,999	100,098
No Alarm Raised	1	9,899,901	9,899,902
Total	100	9,999,900	10,00,00

This example was provided to show the balance we must strike as security professionals when setting the sensitivity of our IDS systems. Such is the main challenge we face today when configuring our IDS's.

Types of IDS's

There are three main types of IDS's:

- **Host-based:** Monitors a single host's activity
    - **Example:** Something that I put onto each of my systems on my network.
- **Network-based:** Monitors network traffic
    - **Example:** Something that sniffs all network traffic and notifies of anomalies, when spotted.
- **Distributed or Hybrid:** Combination of multiple sensors that is typically both host and network based. It then feeds this input to a central analyzer to identify/respond to intrusion activity.

Standard Components of an IDS

While not all IDS's are comprised of the same components, there are a few components which are typically found with using IDS's. These are explored below.

1. **Traffic Collector/Sensor – AKA "a sensor"**
    a. Collects activity/events for the IDS to examine
    b. For hosts, this could be log files, audit logs, or traffic logs from a particular system. Logs can be native OS files, or detection specific audit files.
    c. The sources for which relevant IDS data is produced for analysis are also known as **information (data) sources**.
    d. For networks, this is typically a packet sniffer that copies network traffic is sees
2. **Analysis Engine – AKA "the brain"**
    a. Examines network traffic and compares it to known patterns. Flags malicious activity
3. **Signature Database**
    a. Collection of patterns/definitions of known suspicious/malicious activity
4. **User Interface and Reporting**
    a. Provides alerts when appropriate
    b. Gives users a means to interact with/operate the IDS

Host IDS's (HIDS)

The first type of IDS which we can explore is the host IDS. The software to monitor hosts resides on the specific asset to detect suspicious behavior.

Specific types of data sources which host-based IDS's use include:

- Kernel logs
- Server logs
- Firewall logs
- Other host-specific logs that are produced by the host itself.
- File integrity checksums
- Registry access

HIDS filter through the information sources which reside on the host itself, and create special logs for analysis. Upon detecting an attack, the HIDS can check against an internal database that records common signatures or known attacks.

One reason why someone may want to use a HIDS is if the system itself contains sensitive files. For example, altering a file may show up in network traffic if a user has used *ssh* or *telnet* to connect to the system, but there are also cases where someone might care if a file itself is changed.

HIDS allow users to perform checksum analysis against the files on the host. In the case that a checksum doesn't match the expected output, this too can produce flags.

Network-based IDS (NIDS)

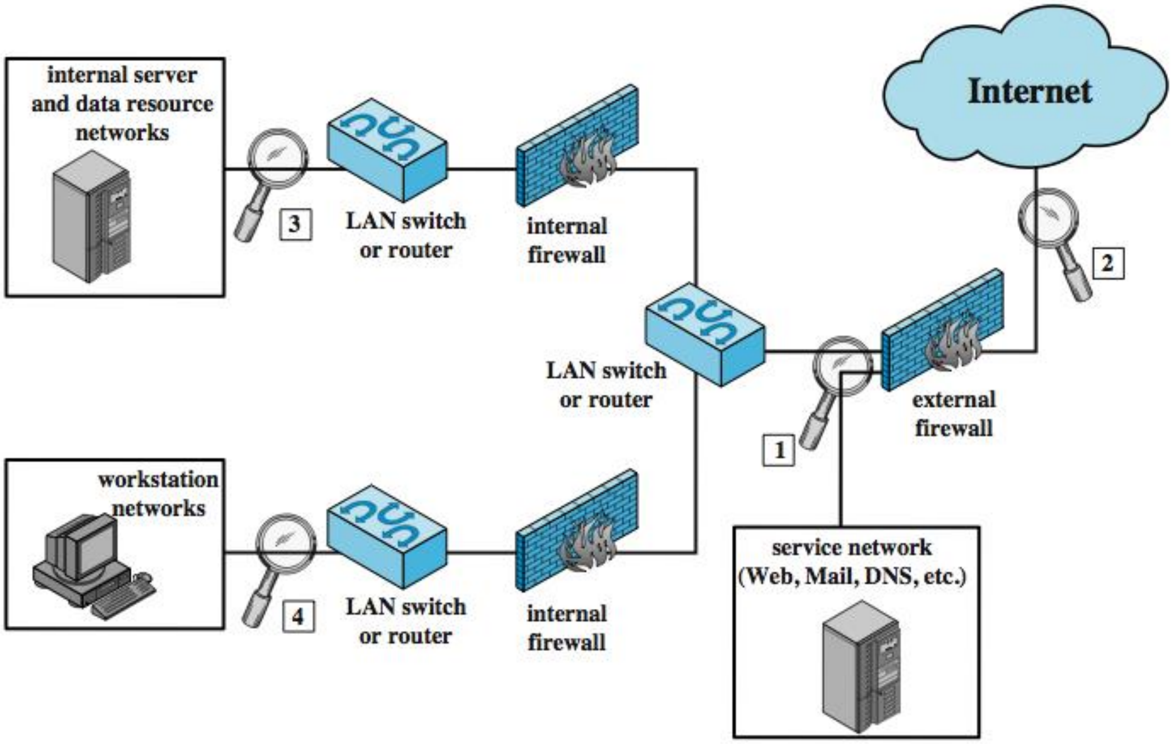
Contrary to what many may think, NIDS don't need to capture *all* the traffic on a network. Rather, NIDS can still be effective when they monitor certain points on a network.

NIDS can be *inline*, where they pass traffic through, or *passive*, where the monitor a copy of the traffic. In either case, once an anomaly is detected at the network, transport, or application protocol level, NIDS can create a corresponding alert for users to investigate the anomaly.

Some information that a NIDS sensor may connect include:

- Timestamps
- Network, transport, and application layer protocols
- Source/Destination IP addresses
- Source/Destination TCP/UDP ports, or ICMP types and codes
- Number of bytes transmitted over a given connection
- Decoded payload data
- State-related information (example: How long a given connection has been established for)

The type of information that is collected changes based on where the NIDS sensor is placed. Below is a picture showing different locations for where one might place a NID sensor:

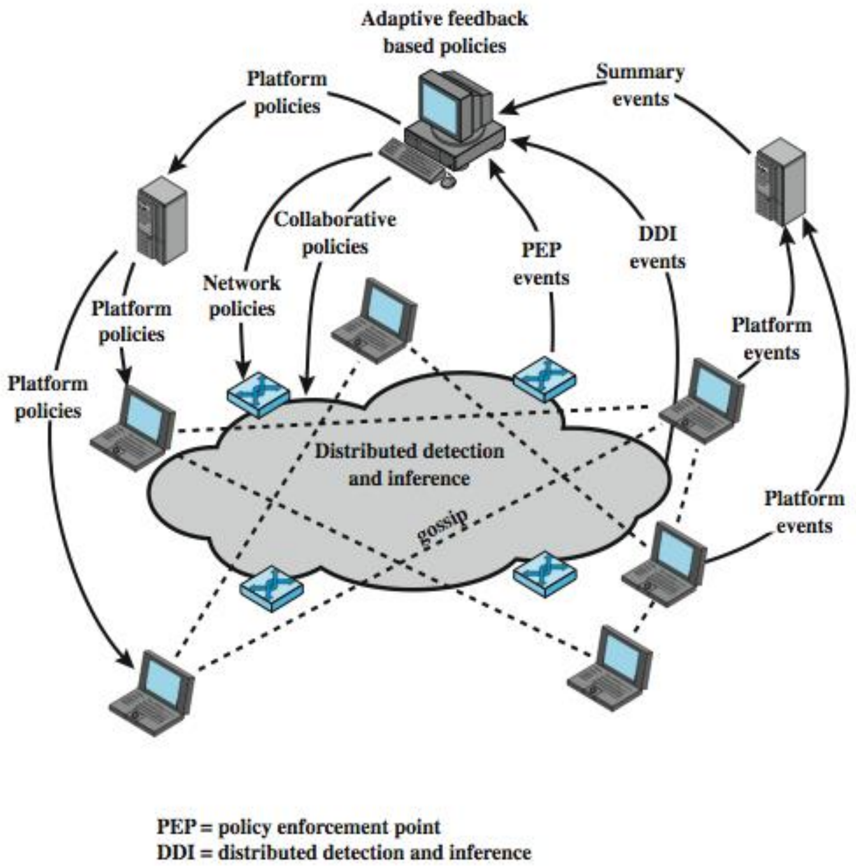


1. Monitors attacks from outside the internal network.
2. Monitors and documents unfiltered traffic. Note: This creates a lot of packets that need to be monitored.
3. Monitors of major internal infrastructure. This detects both internal and external attacks.
4. Monitors traffic to critical systems with sensitive information (example: bank accounts)

Hybrid IDS

As discussed in the previous sections, Hybrid IDS's can also be used to perform detection at both the host and network level. Below is an example of what this might look like on a given network.

Joseph Tsai
Overview of Intrusion Detection Systems



PEP = policy enforcement point
DDI = distributed detection and inference

It may come as no surprise that ideally, hosts and networks are secured as part of an IDS. However, there are challenges when implementing this type of IDS, which include:

1. Needing to deal with multiple audit log formats
2. Needing to decide between a centralized or decentralized architecture for performing analysis and detection
3. Needing a large multitude of sensors to collect data that will feed the analyzer

The diagram above is one potential solution to the challenges described. In the diagram, distribution of policies is done via "gossip" between computers, which allows the policy changes to be dispersed throughout the network. Network traffic is also analyzed as it is sent between hosts on the network to alert of relevant detections.

Design considerations for IDS's

With such complex systems, it is important to consider the multiple facets of designing these IDS's. For example:

1. Deciding on timeliness of an IDS
    a. Will it be "online"? Meaning, will the IDS detect intrusions in real time?
    b. Will it be "offline"? Meaning, will the IDS first store data and then perform analysis on the captured data?

2. Centralized vs. Decentralized
    a. Centralized: Will it only analyze data collected from a single monitoring system?
    b. Distributed: Will it collect data from multiple monitoring systems in order to investigate global and coordinated attacks?
3. How will the analysis be performed?
    a. Will it simply check against well-known attacks?
    b. Will it attempt to define “unusual behavior” and alert based on that?

Types of analysis

Now that we have discussed the high-level components of an IDS, it is also important to discuss how IDS’s perform analysis on the information that they capture.

The first method for analysis is known as **signature** or **misuse** detection. Essentially, there is a set of known data patterns or attack rules. Upon seeing this type of traffic in the captured data, an alert is created.

Examples include:

- Users who are logged in for more than one session
- User that make copies of important files
- Users that write to other people’s files
- Users who log in after hours to access files which they accessed earlier

As with any type of analysis, there are advantages and disadvantages. The main advantage for this type of analysis is that it is very successful at detecting known occurrences of previously known attacks. The disadvantage is that it fails to detect new types of attacks. These new attacks need to be put into the signature database so that they can be detected in the future.

The second main method of analysis used for IDS’s is called **anomaly (behavior) detection**. For this type of analysis, any deviations from standard user behavior will generate an alert. Normal behavior is programmed into the alerting system so that such anomalies can be detected.

Now, there are clearly increased areas of ambiguity with this. For example: “What constitutes normal user behavior? How do we know that the traffic we have is normal to begin with? Do we have enough data that spans a long enough amount of time to establish normality?”

Thankfully, there are *some* instances of abnormality which are more easily detected than others. For example:

- Seeing HTTP traffic on a non-standard port, like port 53.
- Seeing a large spike of UDP traffic when compared to standard levels of TCP traffic.
- Seeing a large number of bytes coming from an HTTP browser than the number of bytes which are being sent to the browser.

To assist with determining normality, it helps to think of what is being used to *profile* the given system. For example, HIDS will have different profiles than NIDS. A HID may use application log files, while a NID may analyze network traffic. The point is, an effective profile and definition of a profile is crucial to establishing a strong IDS.

Within this realm of anomaly detecting IDS', there are a few types of anomaly detection that are frequently used, including:

1. **Statistical anomaly detection** – statistical tests which are used to observe behavior and determine whether or not the system is detecting legitimate user behavior
2. **Threshold detection** – defining thresholds independent of users based on the frequency and occurrence of various events
3. **Profile based** – developing profiles for each user, and detecting changes in user behavior/individual accounts

Special attention should be given to Statistical Anomaly Based Intrusion Detection Systems (SABIDS), as these systems are quite unique in what they have to offer.

Advantages include:

- No needed knowledge of security flaws/the attacks themselves
- Can detect zero day attacks
- Potentially easier to maintain, as they don't rely on specific attacks or conditions

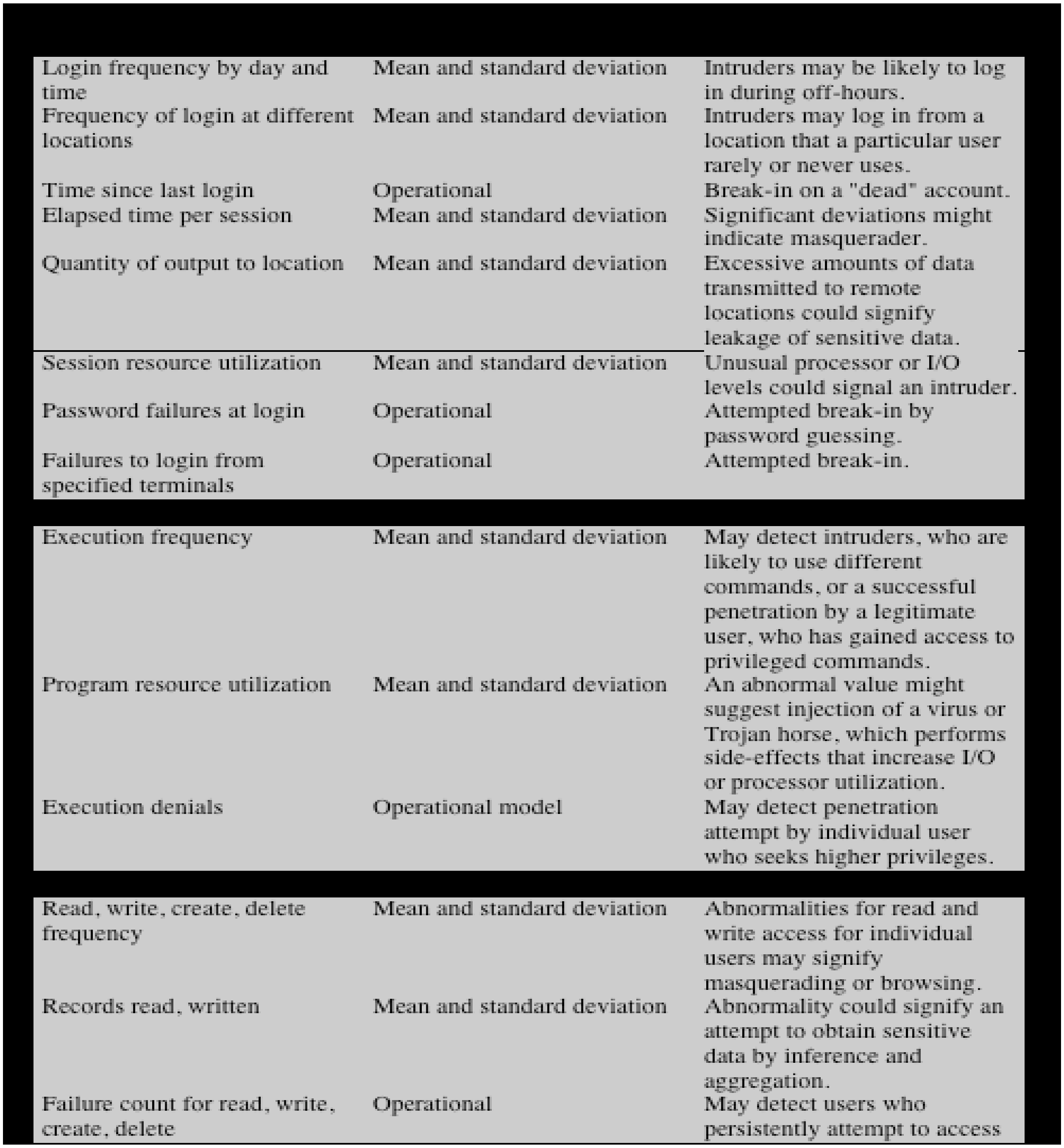
Disadvantages include:

- Some behavior is difficult to model with statistics
- Learning process can take a longer time than signature-based detections
- Fine-tuning of thresholds for alerting can be difficult

There are two main types of SABIDS. At a high level, these are:

1. The operational/threshold model – upon reaching a given threshold, fire an alert
2. Statistical moments or mean/standard deviation model – computes standard deviation across a given period of time for a specified type of event. Having an event fall beyond a given number of standard deviations away from the mean will result an alert being generated.

Here are some measures that can be used for intrusion detection:



Login frequency by day and time	Mean and standard deviation	Intruders may be likely to log in during off-hours.
Frequency of login at different locations	Mean and standard deviation	Intruders may log in from a location that a particular user rarely or never uses.
Time since last login	Operational	Break-in on a "dead" account.
Elapsed time per session	Mean and standard deviation	Significant deviations might indicate masquerader.
Quantity of output to location	Mean and standard deviation	Excessive amounts of data transmitted to remote locations could signify leakage of sensitive data.
Session resource utilization	Mean and standard deviation	Unusual processor or I/O levels could signal an intruder.
Password failures at login	Operational	Attempted break-in by password guessing.
Failures to login from specified terminals	Operational	Attempted break-in.
Execution frequency	Mean and standard deviation	May detect intruders, who are likely to use different commands, or a successful penetration by a legitimate user, who has gained access to privileged commands.
Program resource utilization	Mean and standard deviation	An abnormal value might suggest injection of a virus or Trojan horse, which performs side-effects that increase I/O or processor utilization.
Execution denials	Operational model	May detect penetration attempt by individual user who seeks higher privileges.
Read, write, create, delete frequency	Mean and standard deviation	Abnormalities for read and write access for individual users may signify masquerading or browsing.
Records read, written	Mean and standard deviation	Abnormality could signify an attempt to obtain sensitive data by inference and aggregation.
Failure count for read, write, create, delete	Operational	May detect users who persistently attempt to access

Honeypots as part of IDS's

The last topic this overview will cover is the usage of honeypots.

The purpose of honeypots is to act as decoy systems that lure potential attackers away from critical systems. They do not have production value, and can actually help capture attacker's activity. Some have fabricated information that can be used to keep the attackers on for longer, thus allowing administrators to respond to such attacks.

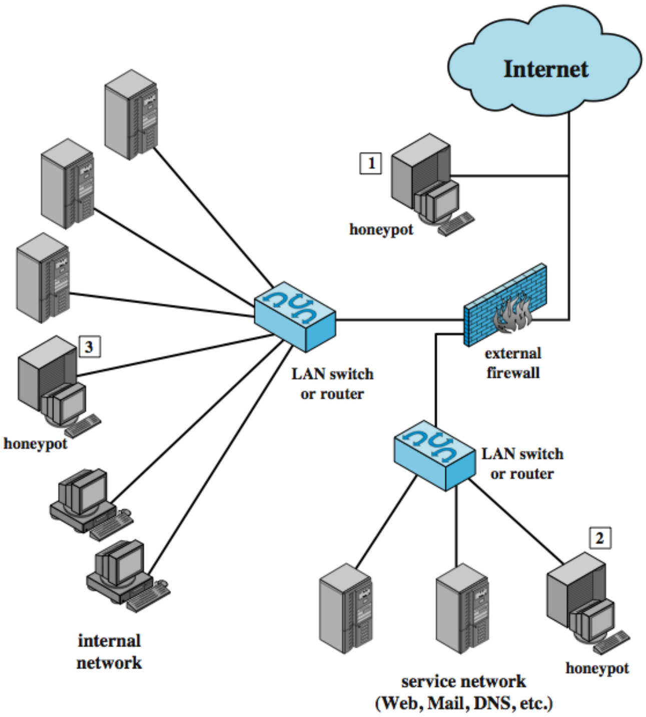
Joseph Tsai
Overview of Intrusion Detection Systems

There are two general versions of honeypots: Low interaction, and high interaction.

Low interaction honeypots consist of software that emulate certain devices. They don't really provide realistic targets, but they serve the purpose of producing warnings in the case of attacks on hosts of on the network.

High interaction systems are real systems with full operating systems and applications. They are intentionally placed in locations that can be accessed by attackers.

The location of where honeypots are deployed also impacts the effectiveness of honeypots. Consider the following diagram:



1. Outside the firewall. This is useful for tracking attempts to connect to unused IP addresses within the scope of the network. Doesn't increase any risk on the internal network, but can't help against insider attacks, either. If there is an external firewall that is filtering traffic, then it has little to no ability to trap internal attackers.
2. In the Demilitarized Zone (DMZ). When we do this, we want to make sure that the other assets in the DMZ are properly hardened. The disadvantage is that the firewall may block a lot of the traffic to the honeypot, so the firewall must be more relaxed in the type of traffic it allows to enter the DMZ, which makes the other systems vulnerable to attacks. There needs to be a balance between risk allowed and having proper firewall rules.
3. The third location is completely internal. Great for insider attacks. Can also detect misconfigured firewalls, if attackers can reach this system. The disadvantage here is that

if the honeypot is compromised to the point where it can attack other systems, then you may be exposing your network to a malicious system that has internal access to the network (traffic isn't blocked by a firewall). Another disadvantage is that you may need to be more relaxed with your firewall rules to allow traffic to come in in order to allow traffic to the honeypot, which complicates firewall configuration and ultimately may lead to the compromising of the internal network.

Conclusion

In conclusion, there are many facets to IDS's, some of which have been explored in this high-level overview. I hope that you have found this reading useful, and look forward to hearing how this can be improved. We have but only studied the tip of a very large iceberg, to say the least.

I wish you all the best in your studying and learning endeavors!

-Joseph