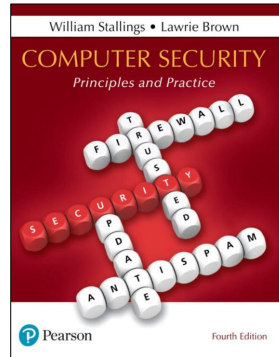




## Computer Security: Principles and Practice



### • Chapter 8

#### • Intrusion Detection



2 Dr. Valerio Formicola

Copyright © 2018, 2015, 2012 Pearson Education, Inc. All Rights Reserved

If this PowerPoint presentation contains mathematical equations, you may need to check that your computer has the following installed:

- 1) MathType Plugin
- 2) Math Player (free versions available)
- 3) NVDA Reader (free versions available)

A significant security problem for networked systems is hostile, or at least unwanted, trespass by users or software. User trespass can take the form of unauthorized logon or other access to a machine or, in the case of an authorized user, acquisition of privileges or performance of actions beyond those that have been authorized. Software trespass includes a range of malware variants as we discuss in Chapter 6.

This chapter covers the subject of intrusions. First, we examine the nature of intruders and how they attack, then look at strategies for detecting intrusions.

## Intrusion and Intrusion Detection

- Security Intrusion:

Unauthorized act of bypassing the security mechanisms of a system

- Intrusion Detection:

A hardware or software function that gathers and analyzes information from various areas within a computer or a network to identify possible security intrusions



CalPoly Pomona

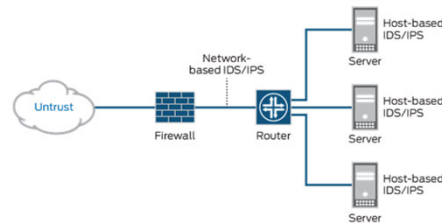
3 Dr. Valerio Formicola

**security intrusion:** Unauthorized act of bypassing the security mechanisms of a system.

**intrusion detection:** A hardware or software function that gathers and analyzes information from various areas within a computer or a network to identify possible security intrusions.

## Intrusion Detection System (IDS)

- **Host-based IDS (HIDS)**
  - Monitors the characteristics of a single host for suspicious activity
- **Network-based IDS (NIDS)**
  - Monitors network traffic and analyzes network, transport, and application protocols to identify suspicious activity
- **Distributed or hybrid IDS**
  - Combines information from a number of sensors, often both host and network based, in a central analyzer that is able to better identify and respond to intrusion activity

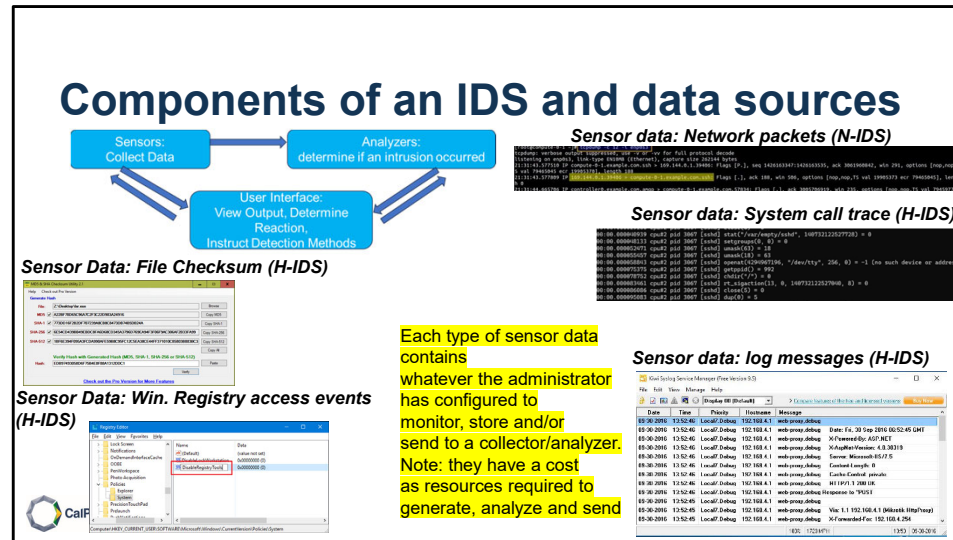


An IDS may use a single sensor and analyzer, such as a classic HIDS on a host or NIDS in a firewall device. More sophisticated IDSs can use multiple sensors, across a range of host and network devices, sending information to a centralized analyzer and user interface in a distributed architecture.

IDSs are often classified based on the source and type of data analyzed, as:

- **Host-based IDS (HIDS):** Monitors the characteristics of a single host and the events occurring within that host, such as process identifiers and the system calls they make, for evidence of suspicious activity.
- **Network-based IDS (NIDS):** Monitors network traffic for particular network segments or devices and analyzes network, transport, and application protocols to identify suspicious activity.
- **Distributed or hybrid IDS:** Combines information from a number of sensors, often both host and

network-based, in a central analyzer that is able to better identify and respond to intrusion activity.



An IDS comprises three logical components:

- **Sensors** : Sensors are responsible for collecting data. The input for a sensor may be any part of a system that could contain evidence of an intrusion. Types of input to a sensor includes **network packets, log files, and system call traces**. Sensors collect and forward this information to the analyzer.
- **Analyzers** : Analyzers receive input from one or more sensors or from other analyzers. The analyzer is responsible for determining if an intrusion has occurred. The output of this component is an indication that an intrusion has occurred. The output may include evidence supporting the conclusion that an intrusion occurred. The analyzer may provide guidance about what actions to take as a result of the intrusion. The sensor inputs may also be stored for future analysis and review in a storage or database component.
- **User interface**: The user interface to an IDS enables a user to view output from the system or control the behavior of the system. In some systems, the user interface may equate to a manager, director, or console

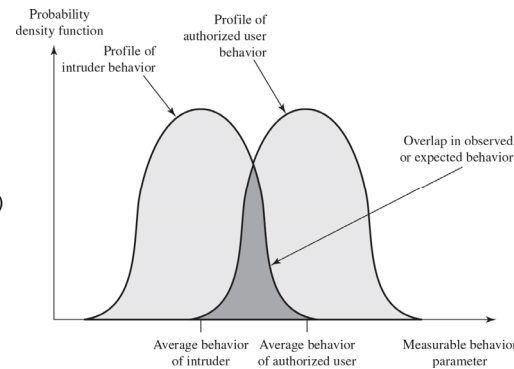
component.

## Profiles of Behavior of Intruders and Authorized Users

### Why is it so hard to detect an intrusion?

Because, except for the cases when Indicators of Compromise are known and detected (malicious known IPs, malicious domains, malware hashes, heuristic rules, etc.)

The behavior of an intruder has many characteristics that correspond to the average user behavior -> it's never a sharp difference



CalPoly Pomona

6 Dr. Valerio Formicola

Authentication facilities, access control facilities, and firewalls all play a role in countering intrusions. Another line of defense is intrusion detection, and this has been the focus of much research in recent years. This interest is motivated by a number of considerations, including the following:

1. If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised. Even if the detection is not sufficiently timely to preempt the intruder, the sooner that the intrusion is detected, the less the amount of damage and the more quickly that recovery can be achieved.
2. An effective IDS can serve as a deterrent, thus acting to prevent intrusions.
3. Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen intrusion prevention measures.



Intrusion detection is based on the assumption that the behavior of the intruder differs from that of a legitimate user in ways that can be quantified. Of course, we cannot expect that there will be a crisp, exact distinction between an attack by an intruder and the normal use of resources by an authorized user. Rather, we must expect that there will be some overlap.

Figure 8.1 suggests, in abstract terms, the nature of the task confronting the designer of an IDS. Although the typical behavior of an intruder differs from the typical behavior of an authorized user, there is an overlap in these behaviors. Thus, a loose interpretation of intruder behavior, which will catch more intruders, will also lead to a number of false positives, or false alarms, where authorized users are identified as intruders. On the other hand, an attempt to limit false positives by a tight interpretation of intruder behavior will lead to an increase in false negatives, or intruders not identified as intruders. Thus, there is an element of compromise and art in the practice of intrusion detection. Ideally you want an IDS to have a high detection rate, that is, the ratio of detected to total attacks, while minimizing the false alarm rate, the ratio of incorrectly classified to total normal usage [LAZA05].

In an important early study of intrusion [ANDE80], Anderson postulated that one could, with reasonable confidence, distinguish between an outside attacker and a legitimate user. Patterns of legitimate user behavior can be established by observing past history, and significant deviation from such patterns can be detected. Anderson suggests that the task of detecting an inside attacker (a legitimate user acting in an unauthorized fashion) is more difficult, in that the distinction between abnormal and normal behavior may be small. Anderson concluded that such violations would be undetectable solely through the search for anomalous behavior. However, insider behavior might nevertheless be detectable by intelligent definition of the class of conditions that suggest unauthorized use. These observations, which were made in 1980, remain true today.

The graph plots probability density function versus measurable behavior parameter. The graph displays two overlapping bell curves. The overlapping region is shaded and labeled, overlap in observed or expected behavior. The bell curve in the left is labeled, profile of intruder behavior and the bell curve in the right is labeled, profile of authorized user behavior. The right tail of the first curve is labeled, average behavior of authorized user. The left tail of the second curve is labeled, average behavior of intruder.

## General approaches for any IDS (host/network/hybrid)

## Anomaly detection

- Involves the collection of data relating to the behavior of legitimate users over a period of time
- Current observed behavior is analyzed to determine whether this behavior is that of a legitimate user or that of an intruder

### Signature/Heuristic detection

- Uses a set of known malicious data patterns or attack rules that are compared with current behavior
- Also known as misuse detection
- Can only identify known attacks for which it has patterns or rules



IDSs typically use one of the following alternative approaches to analyze sensor data to detect intrusions:

1. **Anomaly detection:** Involves the collection of data relating to the behavior of legitimate users over a period of time. Then current observed behavior is analyzed to determine with a high level of confidence whether this behavior is that of a legitimate user or alternatively that of an intruder.

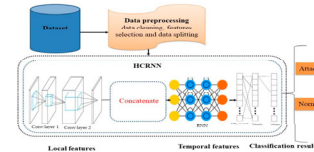
**2. Signature or Heuristic detection:** Uses a set of known malicious data patterns (signatures) or attack rules (heuristics) that are compared with current behavior to decide if is that of an intruder. It is also known as misuse detection. This approach can only identify known attacks for which it has patterns or rules.

In essence, anomaly approaches aim to define normal, or expected, behavior, in order to identify malicious or unauthorized behavior. Signature or heuristic-based approaches directly define malicious or unauthorized behavior. They can quickly and efficiently identify known attacks. However only anomaly detection is able to detect unknown, zero-day attacks, as it starts with known good behavior and identifies anomalies to it.

Given this advantage, clearly anomaly detection would be the preferred approach, were it not for the difficulty in collecting and analyzing the data required, and the high level of false alarms, as we discuss in the following sections.

# Anomaly Detection

- A variety of classification approaches are used:
  - **Statistical**
    - Analysis of the observed behavior using univariate, multivariate, or time-series models of observed metrics
  - **Knowledge-based**
    - Approaches use an expert system that classifies observed behavior according to a set of rules that model legitimate behavior
  - **Machine-learning**
    - Approaches automatically determine a suitable classification model from the training data using data mining techniques



8 Dr. Valerio Formicola

The anomaly detection approach involves first developing a model of legitimate user behavior by collecting and processing sensor data from the normal operation of the monitored system in a training phase. This may occur at distinct times, or there may be a continuous process of monitoring and evolving the model over time. Once this model exists, current observed behavior is compared with the model in order to classify it as either legitimate or anomalous activity in a detection phase.

A variety of classification approaches are used, which [GARC09] broadly categorized as:

- **Statistical:** Analysis of the observed behavior using univariate, multivariate, or time-series models of observed metrics.
- **Knowledge based:** Approaches use an expert system that classifies observed behavior according to a set of rules that model legitimate behavior.

- **Machine-learning:** Approaches automatically determine a suitable classification model from the training data using data mining techniques.

They also note two key issues that affect the relative performance of these alternatives, being the efficiency and cost of the detection process.

The monitored data is first parameterized into desired standard metrics that will then be analyzed. This step ensures that data gathered from a variety of possible sources is provided in standard form for analysis.

Statistical approaches use the captured sensor data to develop a statistical profile of the observed metrics. The earliest approaches used univariate models, where each metric was treated as an independent random variable. However this was too crude to effectively identify intruder behavior. Later, multivariate models considered correlations between the metrics, which better levels of discrimination observed. Time-series models use the order and time between observed events to better classify the behavior. The advantages of these statistical approaches include their relative simplicity and low computation cost, and lack of assumptions about behavior expected. Their disadvantages include the difficulty in selecting suitable metrics to obtain a reasonable balance between false positives and false negatives, and that not all behaviors can be modeled using these approaches.

Knowledge based approaches classify the observed data using a set of rules. These rules are developed during the training phase, usually manually, to characterize the observed training data into distinct classes. Formal tools may be used to describe these rules, such as a finite-state machine or a standard description language. They are then used to classify the observed data in the detection phase. The advantages of knowledge-based approaches include their robustness and flexibility. Their main disadvantage is the difficulty and time required to develop high-quality knowledge from the data, and the need for human experts to assist with this process.

Machine-learning approaches use data mining techniques to automatically develop a model using the labeled normal training data. This model is then able to classify subsequently observed data as either normal or anomalous. A key disadvantage is that this process typically requires significant time and computational resources. Once the model is generated however, subsequent analysis is generally fairly efficient.

A variety of machine-learning approaches have been tried, with varying success. These include:

- **Bayesian networks:** Encode probabilistic relationships among observed metrics.
- **Markov models:** Develop a model with sets of states, some possibly hidden, interconnected by transition probabilities.
- **Neural networks:** Simulate human brain operation with neurons and synapse between them, that classify observed data.
- **Fuzzy logic:** Uses fuzzy set theory where reasoning is approximate, and can accommodate uncertainty.
- **Genetic algorithms:** Uses techniques inspired by evolutionary biology, including inheritance, mutation, selection and recombination, to develop classification rules.
- **Clustering and outlier detection:** Group the observed data into clusters based on some similarity or distance measure, and then identify subsequent data as either belonging to a cluster or as an outlier.

The advantages of the machine-learning approaches include their flexibility, adaptability, and ability to capture interdependencies between the observed metrics. Their disadvantages include their dependency on assumptions about accepted behavior for a system, their currently unacceptably high false alarm rate, and their high resource cost.

A key limitation of anomaly detection approaches used by IDSs, particularly the machine-learning approaches, is that they are generally only trained with legitimate data, unlike many of the other applications surveyed in [CHAN09] where both legitimate and anomalous training data is used. The lack of anomalous training data, which occurs given the desire to detect currently unknown future attacks, limits the effectiveness of some of the techniques listed above.

## Signature or Heuristic Detection

- **Signature approaches**

- Match a large collection of known patterns of malicious data against data stored on a system or in transit over a network
- The signatures need to be large enough to minimize the false alarm rate, while still detecting a sufficiently large fraction of malicious data
- **Widely used in anti-virus products, network traffic scanning proxies, and in NIDS**

- **Rule-based heuristic identification**

- Involves the use of rules for identifying known penetrations or penetrations that would exploit known weaknesses
- Rules can also be defined that identify suspicious behavior, even when the behavior is within the bounds of established patterns of usage
- Typically, rules used are specific
- SNORT is an example of a rule-based NIDS

Snort rule with exact match of content

```
alert udp any any -> any 53 (content:"|01 00 00 01 00 00 00 00 00 01|"; offset:
2; depth: 10; content:"|00 00 29 10 00 00 00 80 00 00 00|"; \
msg: "covert iodine tunnel request"; threshold: type limit, track by_src, count
1, seconds 300; sid: 5619500; rev: 1;)

alert udp any 53 -> any any (content: "|84 00 00 01 00 01 00 00 00 00|"; offset:
2; depth: 10; content:"|00 00 0a 00 01|"; \
msg: "covert iodine tunnel response"; threshold: type limit, track by_src, count
1, seconds 300; sid: 5619501; rev: 1;)
```

Signature or heuristic techniques detect intrusion by observing events in the system and applying either a set of signature patterns to the data, or a set of rules that characterize the data, leading to a decision regarding whether the observed data indicates normal or anomalous behavior.

**Signature approaches** match a large collection of known patterns of malicious data against data stored on a system or in transit over a network. The signatures need to be large enough to minimize the false alarm rate, while still detecting a sufficiently large fraction of malicious data. This approach is widely used in antivirus products, in network traffic scanning proxies, and in NIDS. The advantages of this approach include the relatively low cost in time and resource use, and its wide acceptance. Disadvantages include the significant effort required to constantly identify and review new malware to create signatures able to identify it, and the inability to detect zero-day attacks for which no signatures exist.

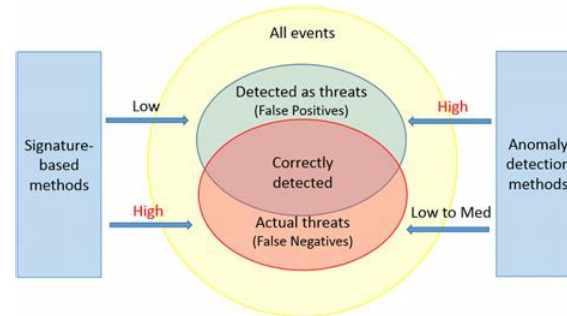
**Rule-based heuristic identification** involves the use of rules for identifying known penetrations or penetrations that would exploit known weaknesses. Rules can also be defined that identify suspicious

behavior, even when the behavior is within the bounds of established patterns of usage. Typically, the rules used in these systems are specific to the machine and operating system. The most fruitful approach to developing such rules is to analyze attack tools and scripts collected on the Internet. These rules can be supplemented with rules generated by knowledgeable security personnel. In this latter case, the normal procedure is to interview system administrators and security analysts to collect a suite of known penetration scenarios and key events that threaten the security of the target system.

The SNORT system, which we discuss later in Section 8.9 is an example of a rule-based NIDS. A large collection of rules exists for it to detect a wide variety of network attacks.



## Signature Vs Anomaly detection



[illegible]

The primary benefit of a HIDS is that it can detect both external and internal intrusions, something that is not possible either with network-based IDSs or firewalls. As we discuss in the previous section, host-based IDSs can use either anomaly or signature and heuristic approaches to detect unauthorized behavior on the monitored host. We now review some common data sources and sensors used in HIDS, and then continue with a discussion of how the anomaly, signature and heuristic approaches are used in HIDS, and then consider distributed HIDS.

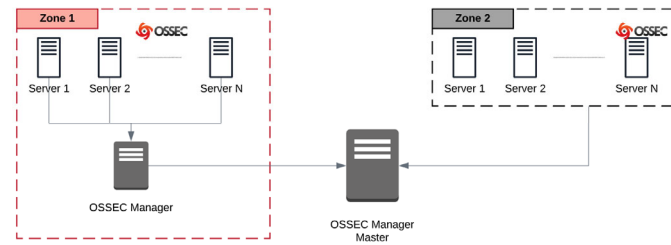
11

record of ongoing activity by users must be provided as input to the analysis component of the IDS. Common data sources include:

- **System call traces:** A record of the sequence of systems calls by processes on a system, is widely acknowledged as the preferred data source for HIDS since the pioneering work of Forrest [CREE13]. While these work well on Unix and Linux systems, they are problematic on Windows systems due to the extensive use of DLLs that obscure which processes use specific system calls.
- **Audit (log file) records:** Most modern operating systems include accounting software that collects information on user activity. The advantage of using this information is that no additional collection software is needed. The disadvantages are that the audit records may not contain the needed information or may not contain it in a convenient form, and that intruders may attempt to manipulate these records to hide their actions.
- **File integrity checksums:** A common approach to detecting intruder activity on a system is to periodically scan critical files for changes from the desired baseline, by comparing a current cryptographic checksums for these files, with a record of known good values. Disadvantages include the need to generate and protect the checksums using known good files, and the difficulty monitoring changing files. Tripwire is a well-known system using this approach.
- **Registry access:** An approach used on Windows systems is to monitor access to the registry, given the amount of information and access to it used by programs on these systems. However this source is very Windows specific, and has recorded limited success.

The sensor gathers data from the chosen source, filters the gathered data to remove any unwanted information and to standardize the information format, and forwards the result to the IDS analyzer, which may be local or remote.

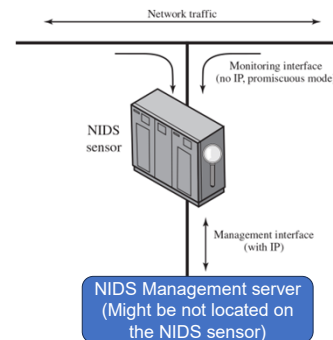
## Example of HIDS: OSSEC



<https://trunc.org/ossec/ossec-as-a-syslog-client-or-server>

## Network-Based IDS (NIDS)

- Monitors traffic at selected points on a network
- Examines traffic packet by packet in real or close to real time
- May examine network, transport, and/or application-level protocol activity
- Comprised of a number of sensors, one or more servers for NIDS management functions, and one or more management consoles for the human interface
- Analysis of traffic patterns may be done at the sensor, the management server or a combination of the two



13 Dr. Valerio Formicola

A network-based IDS (NIDS) monitors traffic at selected points on a network or interconnected set of networks. The NIDS examines the traffic packet by packet in real time, or close to real time, to attempt to detect intrusion patterns. The NIDS may examine network-, transport-, and/or application-level protocol activity. Note the contrast with a host-based IDS; a NIDS examines packet traffic directed toward potentially vulnerable computer systems on a network. A host-based system examines user and software activity on a host.

A typical NIDS facility includes a number of sensors to monitor packet traffic, one or more servers for NIDS management functions, and one or more management consoles for the human interface. The analysis of traffic patterns to detect intrusions may be done at the sensor, at the management server, or some combination of the two.

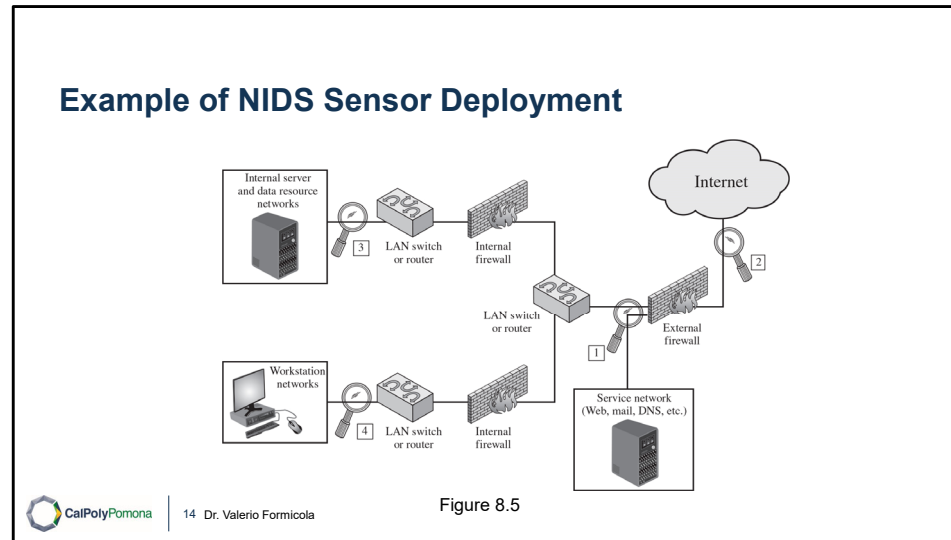
Sensors can be deployed in one of two modes: inline and passive. An inline sensor is inserted into a network segment so that the traffic that it is monitoring must pass through the sensor. One way to achieve

an inline sensor is to combine NIDS sensor logic with another network device, such as a firewall or a LAN switch. This approach has the advantage that no additional separate hardware devices are needed; all that is required is NIDS sensor software. An alternative is a stand-alone inline NIDS sensor. The primary motivation for the use of inline sensors is to enable them to block an attack when one is detected. In this case the device is performing both intrusion detection and intrusion prevention functions.

More commonly, passive sensors are used. A passive sensor monitors a copy of network traffic; the actual traffic does not pass through the device. From the point of view of traffic flow, the passive sensor is more efficient than the inline sensor, because it does not add an extra handling step that contributes to packet delay.

Figure 8.4 illustrates a typical passive sensor configuration. The sensor connects to the network transmission medium, such as a fiber optic cable, by a direct physical tap. The tap provides the sensor with a copy of all network traffic being carried by the medium. The network interface card (NIC) for this tap usually does not have an IP address configured for it. All traffic into this NIC is simply collected with no protocol interaction with the network. The sensor has a second NIC that connects to the network with an IP address and enables the sensor to communicate with a NIDS management server.

The sensor connects to the network transmission medium by a direct physical tap. The tap provides the sensor with a copy of all network traffic being carried by the medium. The monitoring interface for this tap with no I P, promiscuous mode configured for N I D S sensor. The sensor has a management interface that connects to the network with an IP address and enables the sensor to communicate with a N I D S management server.



Consider an organization with multiple sites, each of which has one or more LANs, with all of the networks interconnected via the Internet or some other WAN technology. For a comprehensive NIDS strategy, one or more sensors are needed at each site. Within a single site, a key decision for the security administrator is the placement of the sensors.

Figure 8.5 illustrates a number of possibilities. In general terms, this configuration is typical of larger organizations. All Internet traffic passes through an external firewall that protects the entire facility. Traffic from the outside world, such as customers and vendors that need access to public services, such as Web and mail, is monitored. The external firewall also provides a degree of protection for those parts of the network that should only be accessible by users from other corporate sites. Internal firewalls may also be used to provide more specific protection to certain parts of the network.

A common location for a NIDS sensor is just inside the external firewall ( location 1 in the figure). This position has a number of advantages:

- Sees attacks, originating from the outside world, that penetrate the network's perimeter defenses (external firewall).
- Highlights problems with the network firewall policy or performance.
- Sees attacks that might target the Web server or ftp server.
- Even if the incoming attack is not recognized, the IDS can sometimes recognize the outgoing traffic that results from the compromised server.

Instead of placing a NIDS sensor inside the external firewall, the security administrator may choose to place a NIDS sensor between the external firewall and the Internet or WAN (location 2 ). In this position, the sensor can monitor all network traffic, unfiltered. The advantages of this approach are as follows:

- Documents number of attacks originating on the Internet that target the network
- Documents types of attacks originating on the Internet that target the network

A sensor at location 2 has a higher processing burden than any sensor located elsewhere on the site network.

In addition to a sensor at the boundary of the network, on either side of the external firewall, the administrator may configure a firewall and one or more sensors to protect major backbone networks, such as those that support internal servers and database resources (location 3). The benefits of this placement include the following:

- Monitors a large amount of a network's traffic, thus increasing the possibility of spotting attacks
- Detects unauthorized activity by authorized users within the organization's security perimeter

Thus, a sensor at location 3 is able to monitor for both internal and external attacks. Because the sensor monitors



traffic to only a subset of devices at the site, it can be tuned to specific protocols and attack types, thus reducing the processing burden.

Finally, the network facilities at a site may include separate LANs that support user workstations and servers specific to a single department. The administrator could configure a firewall and NIDS sensor to provide additional protection for all of these networks or target the protection to critical subsystems, such as personnel and financial networks (location 4). A sensor used in this latter fashion provides the following benefits:

- Detects attacks targeting critical systems and resources
- Allows focusing of limited resources to the network assets considered of greatest value

As with a sensor at location 3, a sensor at location 4 can be tuned to specific protocols and attack types, thus reducing the processing burden.

A diagram illustrates the examples of N I D S sensor deployment. All Internet traffic passes through an external firewall. Traffic from the service network such as web, mail, D N S, etcetera are monitored. The external firewall also provides a degree of protection for those parts of the network. A common location for a NIDS sensor is just inside the external firewall, location 1 in the diagram. A N I D S sensor is located between the external firewall and the Internet or WAN, location 2. In addition to a sensor at the boundary of the network, on either side of the external firewall, the administrator configures a firewall and one or more sensors that support internal servers and database resource network at location 3. Finally, the network facilities at a site may include separate L A N's that support user workstations and servers specific to a single department at location 4.

## Intrusion Detection Techniques with NIDS

### Attacks suitable for Signature detection

- Application layer reconnaissance and attacks
- Transport layer reconnaissance and attacks
- Network layer reconnaissance and attacks
- Unexpected application services
- Policy violations

### Attacks suitable for Anomaly detection

- Denial-of-service (DoS) attacks
- Scanning
- Worms
- Zero-days



CalPoly Pomona

15 Dr. Valerio Formicola

As with host-based intrusion detection, network-based intrusion detection makes use of signature detection and anomaly detection. Unlike the case with HIDS, a number of commercial anomaly NIDS products are available [GARC09]. One of the best known is the Statistical Packet Anomaly Detection Engine (SPADE), available as a plug-in for the Snort system that we discuss later.

NIST SP 800-94 (*Guide to Intrusion Detection and Prevention Systems*, July 2012) lists the following as examples of that types of attacks that are suitable for signature detection:

• **Application layer reconnaissance and attacks:** Most NIDS technologies analyze several dozen application protocols. Commonly analyzed ones include Dynamic Host Configuration Protocol (DHCP), DNS, Finger, FTP, HTTP, Internet Message Access Protocol (IMAP), Internet Relay Chat (IRC), Network File System (NFS), Post Office Protocol (POP), rlogin/rsh, Remote Procedure Call (RPC), Session Initiation Protocol (SIP), Server Message Block (SMB), SMTP, SNMP, Telnet, and Trivial File Transfer Protocol (TFTP), as well as database protocols, instant messaging applications, and peer-to-peer file sharing

software. The NIDS is looking for attack patterns that have been identified as targeting these protocols. Examples of attack include buffer overflows, password guessing, and malware transmission.

- **Transport layer reconnaissance and attacks:** NIDSs analyze TCP and UDP traffic and perhaps other transport layer protocols. Examples of attacks are unusual packet fragmentation, scans for vulnerable ports, and TCP-specific attacks such as SYN floods.

- **Network layer reconnaissance and attacks:** NIDSs typically analyze IPv4, IPv6, ICMP, and IGMP at this level. Examples of attacks are spoofed IP addresses and illegal IP header values.

- **Unexpected application services:** The NIDS attempts to determine if the activity on a transport connection is consistent with the expected application protocol. An example is a host running an unauthorized application service.

- **Policy violations:** Examples include use of inappropriate Web sites and use of forbidden application protocols.

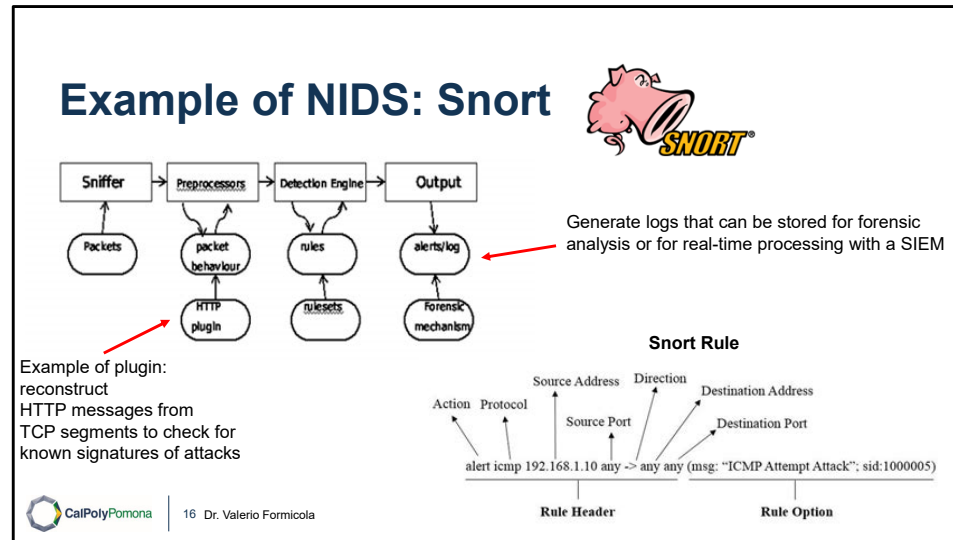
NIST SP 800-94 lists the following as examples of the types of attacks that are suitable for anomaly detection:

- **Denial-of-service (DoS) attacks:** Such attacks involve either significantly increased packet traffic or significantly increase connection attempts, in an attempt to overwhelm the target system. These attacks are analyzed in Chapter 7. Anomaly detection is well suited to such attacks.

- **Scanning:** A scanning attack occurs when an attacker probes a target network or system by sending different kinds of packets. Using the responses received from the target, the attacker can learn many of the system's characteristics and vulnerabilities. Thus, a scanning attack acts as a target identification tool for an attacker. Scanning can be detected by atypical flow patterns at the application layer (e.g., banner grabbing<sup>3</sup>), transport layer (e.g., TCP and UDP port scanning), and network layer (e.g., ICMP scanning).

- **Worms:** Worms spreading among hosts can be detected in more than one way. Some worms propagate quickly and use large amounts of bandwidth. Worms can also be detected because they can cause hosts to communicate

with each other that typically do not, and they can also cause hosts to use ports that they normally do not use. Many worms also perform scanning. Chapter 6 discusses worms in detail.



Snort is an open source, highly configurable and portable host-based or network-based IDS. Snort is referred to as a lightweight IDS, which has the following characteristics:

- Easily deployed on most nodes (host, server, router) of a network
- Efficient operation that uses small amount of memory and processor time
- Easily configured by system administrators who need to implement a specific security solution in a short amount of time

Snort can perform real-time packet capture, protocol analysis, and content searching and matching. Snort can detect a variety of attacks and probes, based on a set of rules configured by a system administrator.

A Snort installation consists of four logical components ( Figure 8.9 ):

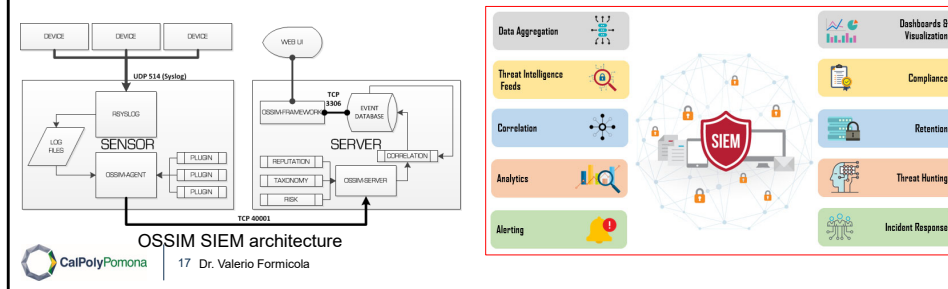
- Packet decoder: The packet decoder processes each captured packet to identify and isolate protocol headers at the data link, network, transport, and application layers. The decoder is designed to be as efficient as possible and its primary work consists of setting pointers so that the various protocol headers can be easily extracted.
- Detection engine: The detection engine does the actual work of intrusion detection. This module analyzes each packet based on a set of rules defined for this configuration of Snort by the security administrator. In essence, each packet is checked against all the rules to determine if the packet matches the characteristics defined by a rule. The first rule that matches the decoded packet triggers the action specified by the rule. If no rule matches the packet, the detection engine discards the packet.
- Logger: For each packet that matches a rule, the rule specifies what logging and alerting options are to be taken. When a logger option is selected, the logger stores the detected packet in human readable format or in a more compact binary format in a designated log file. The security administrator can then use the log file for later analysis.
- Alerter: For each detected packet, an alert can be sent. The alert option in the matching rule determines what information is included in the event notification. The event notification can be sent to a file, to a UNIX socket, or to a database. Alerting may also be turned off during testing or penetration studies. Using the UNIX socket, the alert can be sent to a management machine elsewhere on the network.

A Snort implementation can be configured as a passive sensor, which monitors traffic but is not in the main transmission path of the traffic, or an inline sensor, through which all packet traffic must pass. In the latter case, Snort can perform intrusion prevention as well as intrusion detection. We defer a discussion of intrusion prevention to Chapter 9.

The flow diagram is as follows. Step 1. Packet. Step 2. Decoder. Step 3. Detection engine. The detection engine provides log and alert message.

# Security Information and Event Management (SIEM)

- A Security Information and Event Management (SIEM) system is a distributed system that collects, correlates, analyze and stores logs (aka, *events*) and metrics from various sensors to prevent or detect intrusions
- **Sensors** are sometime called **probes**, and can be NIDS, HIDS, antivirus, firewalls, and any source of information useful to spot an attack (mostly, generating logs collected by the SIEM)



Most common capabilities of a SIEM:

**Data aggregation:** [Log management](#) aggregates data from many sources, including networks, security, servers, databases, applications, providing the ability to consolidate monitored data to help avoid missing crucial events.

•**Correlation:** Looks for common attributes and links events together into meaningful bundles. This technology provides the ability to perform a variety of correlation techniques to integrate different sources, in order to turn data into useful information. Correlation is typically a function of the Security Event Management portion of a full SIEM solution

•**Alerting:** The automated analysis of correlated events

•**Dashboards:** Tools can take event data and turn it into informational charts to assist in seeing patterns, or identifying activity that is not forming a standard pattern.

•**Compliance:** Applications can be employed to automate the gathering of compliance data, producing reports that adapt to existing security, governance and auditing processes.

•**Retention:** Employing long-term storage of historical data to facilitate correlation of data over time, and to

provide the retention necessary for compliance requirements. The Long term log [data retention](#) is critical in forensic investigations as it is unlikely that the discovery of a network breach will be at the time of the breach occurring.

•**Forensic analysis:** The ability to search across logs on different nodes and time periods based on specific criteria. This mitigates having to aggregate log information in your head or having to search through thousands and thousands of logs.

#### Generic Operation of a SIEM:

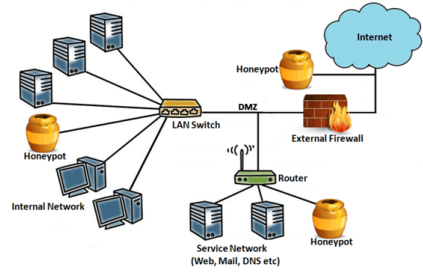
SIEM architectures may vary by vendor; however, generally, essential components comprise the SIEM engine. The essential components of a SIEM are as follows:[\[30\]](#)

- A data collector forwards selected audit logs from a host (agent based or host-based log streaming into index and aggregation point) [\[31\]](#)[\[32\]](#)
- An ingest and indexing point aggregation point for parsing, correlation, and data normalization [\[33\]](#)
- A search node that is used for visualization, queries, reports, and alerts (analysis take place on a search node) [\[34\]](#)



# Honeypot

- Decoy systems designed to:
  - Lure a potential attacker away from critical systems
  - Collect information about the attacker's activity
  - Encourage the attacker to stay on the system long enough for administrators to respond
- Systems are filled with fabricated information that a legitimate user of the system wouldn't access
- **To be effective in the study of the attackers, a honeypot needs to be instrumented as much as possible with any kinds of sensors and probes**
- Two categories of Honeypot:
  - **Low interaction honeypot:** Consists of a software package that emulates particular IT services or systems well enough to provide a realistic initial interaction but does not execute a full version of those services or systems.
  - **High interaction honeypot:** Is a real system, with a full operating system, services and applications, which are instrumented and deployed where they can be accessed by attackers.



Example:  
<https://www.dshield.org/honeypot.html>  
 Installation on Rpi:  
<https://medium.com/swlh/installing-dshield-honeypot-on-a-raspberry-pi-e10d967825b2>

CalPolyPomona | 18 Dr. Valerio Formicola

Honeybots are typically classified as being either low or high interaction.

- **Low interaction honeypot:** Consists of a software package that emulates particular IT services or systems well enough to provide a realistic initial interaction, but does not execute a full version of those services or systems.

- **High interaction honeypot:** Is a real system, with a full operating system, services and applications, which are instrumented and deployed where they can be accessed by attackers.

A high interaction honeypot is a more realistic target that may occupy an attacker for an extended period. However, it requires significantly more resources, and if compromised could be used to initiate attacks on other systems. This may result in unwanted legal or reputational issues for the organization running it. A low interaction honeypot provides a less realistic target, able to identify intruders using the earlier stages of the attack methodology we discussed earlier in this chapter. This is often sufficient for use as a component of a

distributed IDS to warn of imminent attack. “The HoneyNet Project” provides a range of resources and packages for such systems.

Initial efforts involved a single honeypot computer with IP addresses designed to attract hackers. More recent research has focused on building entire honeypot networks that emulate an enterprise, possibly with actual or simulated traffic and data. Once hackers are within the network, administrators can observe their behavior in detail and figure out defenses.