

Module 12: Evading IDS, Firewalls, and Honeypots

Introduction

The widespread use of the internet in the business world has significantly increased overall network usage. Organizations implement various network security measures, including firewalls, Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), and honeypots, to safeguard their networks. Networks are prime targets for hackers aiming to compromise an organization's security, and attackers continually develop new methods to evade these security measures.

In this chapter, you will explore:

- Concepts of IDS, IPS, and firewalls
- Different IDS, IPS, and firewall solutions
- Various techniques used to bypass IDS
- Techniques employed to bypass firewalls
- Methods to bypass Network Access Control (NAC) and endpoint security
- Tools for evading IDS and firewalls
- Honeypot concepts and identify techniques to detect honeypots
- Countermeasures against IDS and firewall evasion

IDS, IPS, and Firewall Concepts

Ethical hackers must possess a comprehensive understanding of the functions, roles, placements, and designs of firewalls, Intrusion Detection Systems (IDS), and Intrusion Prevention Systems (IPS). They need to be aware of the methods employed by attackers to bypass these security measures. This section provides an overview of these critical concepts.

Intrusion Detection System (IDS)

The main differentiation between IPS and IDS is the placement of sensors within a network. A sensor can be placed in line with the network, i.e., the common in/out of a specific network segment terminates on a sensor's hardware or logical interface and goes out from a sensor's second piece of hardware or logical interface. In this situation, every single packet will be analyzed and then only pass through the sensor if it does not contain anything malicious. The trusted network or network segment is protected from known threats and attacks by filtering out malicious traffic. This is the basics of an Intrusion Prevention System (IPS). However, the inline installation and inspection of traffic may result in a slight delay. It is also possible for IPS to become a single point of failure for the whole network. If fail-open mode is used, both the good and the malicious traffic will pass the IPS sensor if it fails in any way. Similarly, if the fail-close mode is configured, the whole IP traffic will be dropped if the sensor fails.

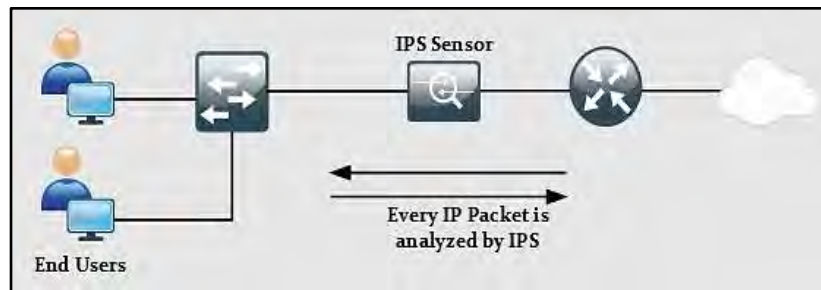


Figure 12-01: In-Line Deployment of IPS Sensor

If a sensor is installed in the position shown in figure 12-02, a copy of every packet will be sent to the sensor to analyze any malicious activity.

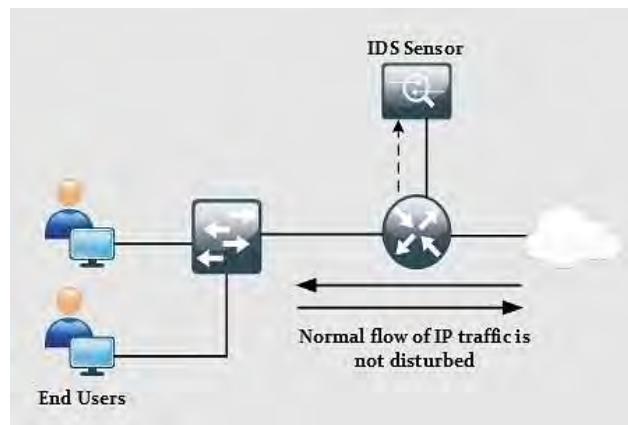


Figure 12-02: Sensor Deployment as IDS

In other words, a sensor running in promiscuous mode will perform the detection and generate an alert if required. As the normal flow of traffic is not disturbed, no end-to-end delay is introduced by implementing IDS. The only downside to this configuration is that IDS cannot stop malicious packets from entering the network because IDS does not control the overall path of traffic. Table 12-01 summarizes and compares various features of IDS and IPS.

Feature	IPS	IDS
Positioning	In-line with the network. Every packet goes through it	Not in-line with the network. It receives a copy of every packet
Mode	In-line/Tap	Promiscuous
Delay	Introduces delay because every packet is analyzed before forwarded to the destination	Does not introduce delay because it is not in-line with the network
Point of Failure?	Yes. If the sensor is down, it may drop or prevent malicious traffic from entering the network, depending on the mode configured on it, namely, fail-open or fail-close	No. Impact on traffic as IDS is not in-line with the network
Ability to Mitigate an Attack?	Yes. By dropping the malicious traffic, attacks can be readily reduced on the network. If deployed in TAP mode, then it will get a copy of each packet but cannot mitigate the attack	IDS cannot directly stop an attack. However, it assists some in-line devices like IPS to drop certain traffic to stop an attack
Can Packets do manipulation?	Yes. Can modify the IP traffic according to a defined set of rules	No. As IDS receives mirrored traffic, it can only perform the inspection

Table 12-01: IDS/IPS Comparison

Intrusion Prevention System (IPS)

Intrusion Prevention Systems (IPS) are active IDS that not only detect but also prevent intrusions. They continuously monitor network traffic while positioned behind firewalls, operating inline to analyze and make automated decisions. Key functions of an IPS include:

- Generating alerts for abnormal traffic
- Recording real-time logs of network activities
- Blocking and filtering malicious traffic
- Quickly detecting and eliminating threats
- Accurately identifying threats without generating false positives

IPS operates based on configured rules and policies, allowing them to log and prevent intrusions and address issues like insider threats and malicious network guests.

Like IDS, IPS are also classified into two types:

1. Host-based IPS
2. Network-based IPS

Unlike Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS) can block and drop illegal packets in the network. IPS can monitor activities within a single organization and prevent direct attacks by managing the volume of network traffic.

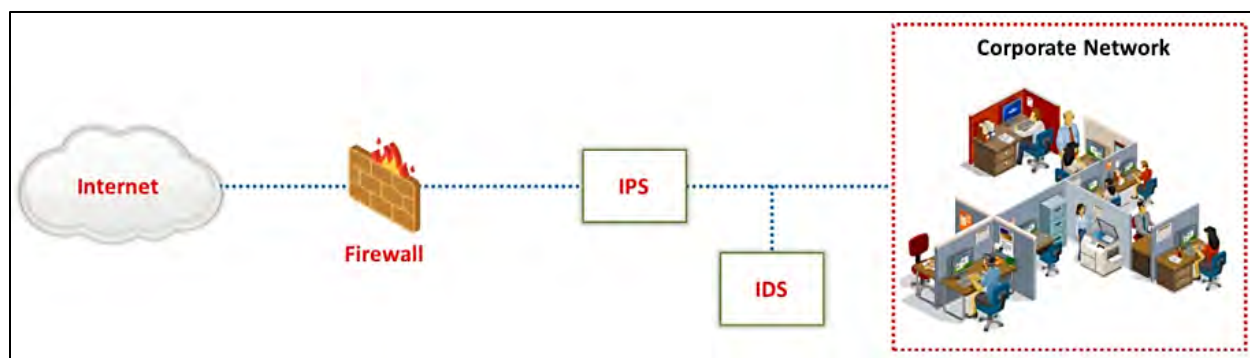


Figure 12-03: Example of an IPS Placement

How an IDS Detects an Intrusion?

When a sensor is analyzing traffic for something strange, it uses multiple techniques based on the rules defined in the IPS/IDS sensor. The following tools and techniques can be used in this regard:

- Signature-based IDS/IPS
- Policy-based IDS/IPS
- Anomaly-based IDS/IPS
- Reputation-based IDS/IPS

Signature-based IDS/IPS

A signature detects an anomaly by looking for some specific string or behavior in a single packet or stream of packets. Cisco IPS/IDS and next-generation firewalls come with pre-loaded digital signatures, which can be used to mitigate previously discovered attacks. Cisco constantly updates the signature set, which also needs to be uploaded to a device by the network administrator.

Not all signatures are enabled by default. If a signature generates false positive alerts, that is, alerts for legitimate traffic, the network administrator needs to tune the IPS/IDS to reduce them.

Policy-based IDS/IPS

Policy-based IDS/IPS are based on an organization's policy or Standard Operating Procedure (SOP). For example, if an organization has a security policy, then no management session using networking devices or end devices can initiate it via the TELNET protocol. A custom rule specifying this policy needs to be defined for sensors. If the rule is configured on IPS, an alert will be generated whenever TELNET traffic hits the IPS, followed by the packets being dropped. If it is implemented on an IDS-based sensor, an alert will be generated for it, but the traffic will keep flowing because IDS works in promiscuous mode.

Anomaly-based IDS/IPS

In anomaly-based IDS/IPS, a baseline is created for specific kinds of traffic. Take, for example, a situation where, after analyzing the traffic, it is noticed that 30 half-open TCP sessions are created every minute. A baseline of 35 half-open TCP connections a minute is set. Assume that the number of half-open TCP connections rises to 150. Based on this anomaly, IPS will drop the extra half-open connections and generate an alert for them.

Reputation-based IDS/IPS

Reputation-based IDS/IPS is useful if there is some global attack, for example, the recent DDoS attacks on Twitter servers and some other social websites. In this situation, it would be useful to filter out the traffic that results from these attacks before it hits the organization's critical infrastructure. Reputation-based IDS/IPS collects information from systems that participate in global correlation. Reputation-based IDS/IPS includes relative descriptors such as known URLs, domain names, etc. Cisco Cloud Services maintains global correlation services.

Table 12-02 summarizes the different technologies used in IDS/IPS, along with some advantages and disadvantages.

IDS/IPS Technology	Advantages	Disadvantages
Signature-based	Easier Implementation and management	Does not detect attacks that can bypass the signatures. May require some tweaking to stop generating false positives for legitimate traffic
Anomaly-based	Can detect malicious traffic based on the custom baseline. It can deny any kind of latest attack, as they are not defined within the scope of baseline policy	Requires baseline policy. It is difficult to baseline large network designs. It may generate false positive alerts due to a misconfigured baseline
Policy-Based	This is a simple implementation with reliable results. Everything else outside the scope of the defined policy is dropped.	This requires manual implementation of policy. Any slight change within a network will also require a change in policy that is configured in IPS/IDS module
Reputation-based	Uses information provided by Cisco Cloud Services in which systems share their experience of network attacks. One person's experience becomes a protection method for other organizations	Requires regular updates and participation in Cisco Cloud Services for global correlation, in which systems share their experience with other members

Table 12-02: Comparison of Techniques Used by IDS/IPS Sensors

General Indications of Intrusions

Intrusion attempts on networks, systems, or file systems can be recognized by observing certain common indicators.

File System Intrusions

By examining system files, one can detect intrusions. System files record system activities, so any changes or deletions suggest a potential attack:

- Unfamiliar files or programs may indicate the system has been compromised, putting other network systems at risk
- Attackers often seek to escalate their privileges to gain administrative control, allowing them to modify file permissions
- Unexpected changes in file size, ownership, or permissions can signal an attack, necessitating a thorough file analysis
- Unauthorized suid and sgid files on Linux that do not match the master list may point to an intrusion
- Look for unfamiliar file names, including executables with odd or double extensions
- Missing files can indicate a possible attack
- Unexplained disk space usage or sudden storage loss is concerning
- Abnormal system behavior, like slow performance or frequent crashes, may also be a sign of an issue
- If an attacker accesses a file system, it could lead to reduced bandwidth due to high resource consumption.

Network Intrusions

Similarly, general indications of network intrusions include:

- An abrupt rise in bandwidth usage
- Continuous scanning of available services on your devices
- Connection attempts from IP addresses outside the expected network range indicate that an unauthorized user (intruder) is trying to access the network
- Numerous login attempts from external hosts
- A sudden surge in log data, which may suggest attempts at DoS attacks, high bandwidth utilization, and DDoS assaults
- Unanticipated modifications in network settings or firewall configurations
- Unexpected system crashes or slowdowns due to increased network activity
- Uncommon outgoing connections or traffic directed toward malicious sites

System Intrusions

Similarly, general indications of system intrusions include:

- Sudden changes in log files, such as incomplete entries or shortened logs
- Unusually slow system performance
- Missing logs or logs that have incorrect permissions or ownership
- Modifications to system software and configuration files
- Unusual graphic displays or unexpected text messages
- Gaps in system accounting records
- System crashes or unexpected reboots
- Unfamiliar processes running on the system
- Alerts from intrusion detection or antivirus software
- Installation of unauthorized software or applications on the system
- Presence of artifacts, such as shell history files, temporary files, or remnants of attacker tools

Types of Intrusion Detection Systems

Depending on the network scenario, IDS/IPS are deployed in one of the following configurations:

1. Host-based Intrusion Detection
2. Network-based Intrusion Detection

Host-based IPS/IDS is normally deployed to protect a specific host machine. It works closely with that machine's Operating System Kernel, creating a filtering layer that blocks any malicious application calls to the OS.

There are four major types of Host-based IDS/IPS:

- **File System Monitoring:** In this configuration, IDS/IPS works by closely comparing the versions of files within a directory with the previous versions of the same files and checks for any unauthorized tampering or changes within the files. Hashing algorithms are often used to verify the integrity of files and directories that indicate possible changes have occurred.
- **Log Files Analysis:** In this configuration, IDS/IPS analyzes the host machine's log files and generates a warning for the system's administrators responsible for machine security. Several tools and applications are available that analyze behavior patterns and further correlate them with actual events.
- **Connection Analysis:** IDS/IPS works by monitoring the overall network connections being made with the secure machine and determining which are legitimate and how many are unauthorized. Examples of techniques used are open port scanning, half-open and rogue TCP connections, and so forth.
- **Kernel Level Detection:** In this configuration, the OS kernel itself detects changes within the system binaries, and any anomaly in the system alerts it to detect intrusion attempts on that machine.

The network-based IPS solution works in line with a perimeter edge device or a specific segment of the overall network. As a network-based solution works by monitoring the overall network traffic (specifically, data packets), it should be as fast as possible in terms of processing power so that overall latency is not introduced to the network. Which technology an IDS/IPS uses depends on the vendor and series.

Feature	Host-based IDS/IPS	Network-based IDS/IPS
Scalability	Not scalable as the number of secure hosts increases	Highly scalable. Normally deployed at perimeter gateway
Cost-Effectiveness	Low. More systems mean more IDS/IPS modules	High. One pair can monitor the overall network
Capability	Capable of verifying whether an attack succeeded or not	Only capable of generating an alert of an attack
Processing Power	The processing power of the host device is used	Must have high processing power to overcome latency issues

Table 12-03: Host-based vs. Network-based IDS/IPS Solutions

Types of IDS Alerts

An Intrusion Detection System (IDS) produces four categories of alerts: True Positive, False Positive, False Negative, and True Negative.

- 1. True Positive (Attack - Alert):** A true positive occurs when an event sets off an alarm, prompting the IDS to act as though a genuine attack is underway. This event could be an actual attack, where an intruder tries to infiltrate the network, or it could be a drill, where security personnel use hacker tools to test a particular network segment.
- 2. False Positive (No attack - Alert):** A false positive happens when an event triggers an alarm without any real attack taking place. This situation arises when an IDS misinterprets normal system activity as an attack. False positives can desensitize users towards alerts, ultimately diminishing their response to real intrusion incidents. During the testing phase of an IDS configuration, administrators leverage false positives to assess if the IDS can differentiate between false alarms and actual threats.
- 3. False Negative (Attack - No Alert):** A false negative is a situation where an IDS does not respond to a genuine attack event. This type of failure is the most critical because the primary goal of an IDS is to identify and react to attacks.
- 4. True Negative (No attack - No Alert):** A true negative occurs when an IDS recognizes a particular action as legitimate behavior and, indeed, that behavior is acceptable. A true negative indicates that the IDS effectively disregards acceptable activity. This scenario poses no threat, signifying that the IDS is functioning correctly in this instance.

Firewall

The primary function of using a dedicated firewall at the edge of a corporate network is isolation. A firewall prevents the internal LAN from having a direct connection with the internet or the outside world. This isolation is carried out by a firewall but is not limited to:

- A Layer 3 device using an Access List for restricting the specific type of traffic on any of its interfaces
- A Layer 2 device using the concept of VLANs or Private VLANs (PVLAN) for separating the traffic of two or more networks
- A dedicated host device with the installed software. This host device, also acting as a proxy, filters the desired traffic while allowing the remaining traffic

Although the features above provide isolation in some sense, the following are reasons for preferring a dedicated firewall appliance (either in hardware or in software) in production environments:

Risks	Protection by firewall
Access by Untrusted Entities	Firewalls try to categorize the network into different portions. One portion is the trusted portion of internal LAN. Public internet interfaces are considered untrusted. Similarly, servers accessed by untrusted entities are placed in a special segment known as a Demilitarized Zone (DMZ). By allowing only specific access to these servers, like port 90 of the webserver, firewalls hide the functionality of a network device, making it difficult for an attacker to understand the physical topology of the network
Deep Packet Inspection and Protocol Exploitation	One of the interesting features of a dedicated firewall is its ability to inspect traffic at more than just IP and port levels. By using digital certificates, Next-Generation Firewalls that are available today can inspect traffic up to layer 7. A firewall can also limit the number of established as well as half-open TCP/UDP connections to mitigate DDoS attacks
Access Control	By implementing local AAA or by using ACS/ISE servers, the firewall can permit traffic based on AAA policy
Anti-virus and Protection from Infected Data	By integrating IPS/IDP modules with a firewall, malicious data can be detected and filtered at the edge of the network to protect end-users

Table 12-04: Firewall Risk Mitigation Features

Although a firewall provides great security features, as discussed in Table 12-04, any misconfiguration or bad network design may result in serious consequences. Another important deciding factor when deploying a firewall in the current network design is whether the current business objectives can bear the following limitations:

- **Misconfiguration and its Consequences:** The primary function of a firewall is to protect network infrastructure in a more elegant way than a traditional layer 3/2 device. Depending on the vendor and their implementation techniques, many features need to be configured for a firewall to work properly. Some of these features may include Network Address

Translation (NAT), Access-Lists (ACL), AAA base policies, and so on. Misconfiguration of any of these features may result in the leakage of digital assets, which may have a financial impact on the business. In short, complex devices like firewalls require deep insight and knowledge of equipment along with the general approach to deployment.

- **Applications and Services Support:** Most firewalls use different techniques to mitigate advanced attacks. For example, NATing, one of the most commonly used features in firewalls, is used to mitigate reconnaissance attacks. In situations where network infrastructure is used to support custom-made applications, it may be necessary to re-write the whole application in order for it to work properly under the new network changes.
- **Latency:** Just as implementing NATing on a route adds some end-to-end delay, a firewall, along with heavy processing demands, can add a noticeable delay to the network. Applications like Voice Over IP (VOIP) may require special configurations to deal with this.

Firewall Architecture

An important factor to consider when designing a network infrastructure's security policies is using the layered approach instead of relying on a single element. Consider the scenario mentioned in Figure 12-04.

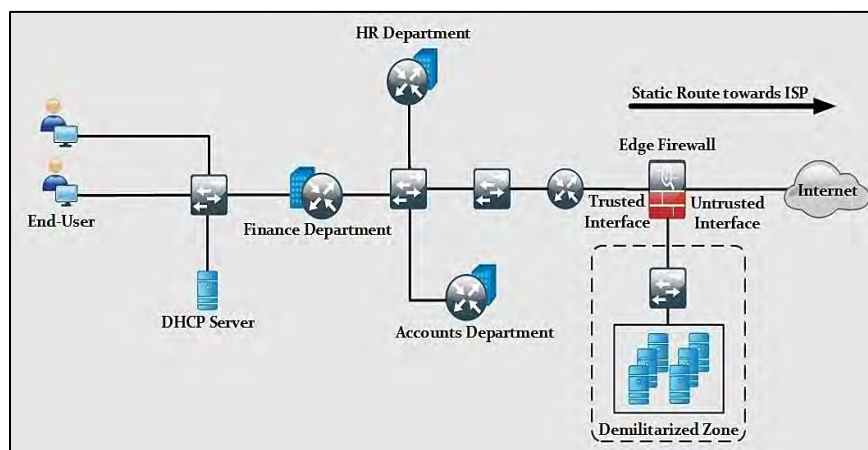


Figure 12-04: Positioning a Firewall in a Production Environment

Figure 12-04 shows a typical scenario of a Small Office Home Office (SOHO) and mid-sized corporate environments where a couple of routers and switches support the whole network infrastructure. If the edge firewall is supposed to be the focal point of security implementation, then any slight misconfiguration may result in high-scale attacks. In general, a layered security approach is followed, and packets pass through multiple security checks before hitting the intended destination.

The position of a firewall varies in different designs. In some designs, it is placed on the corporation's perimeter router, while in other designs, it is placed at the edge of the network, as shown in Figure 12-04. Apart from the position, it is good practice to implement layered security, in which some features, such as unicast reverse path forwarding, access-lists, etc., are enabled on

the perimeter router. Features such as deep packet inspection and digital signatures are matched on the firewall. If everything looks good, the packet is allowed to hit the intended destination address.

Network layer firewalls permit or drop IP traffic based on Layer 3 and 4 information. A router with an access list configured on its interfaces is a common example of a network layer firewall. Although they operate very fast, network layer firewalls do not perform deep packet inspection techniques or detect any malicious activity.

Apart from acting as the first line of defense, network layer firewalls are also deployed within internal LAN segments for enhanced layered security and isolation.

Bastion Host

A Bastion Host is a computer system placed between public and private networks. It is intended to be a crossing point through which traffic passes. The system is assigned certain roles and responsibilities. A bastion host has two interfaces, one connected to the public network and the other to a private network.

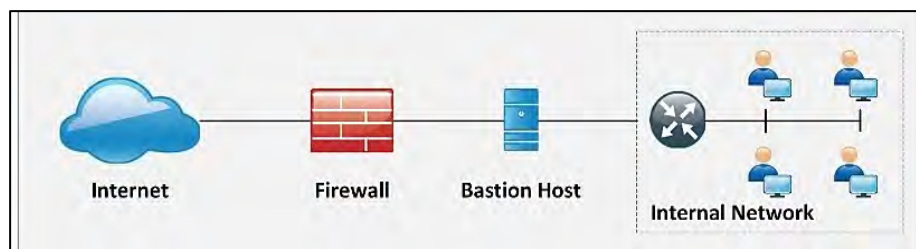


Figure 12-05: Bastion Host

Screened Subnet

Screened Subnet can be set up with a firewall with three interfaces. These three interfaces are connected with the internal Private Network, Public Network, and Demilitarized Zone (DMZ). In this architecture, each zone is separated by another zone; hence, any compromise of one zone will not affect another.

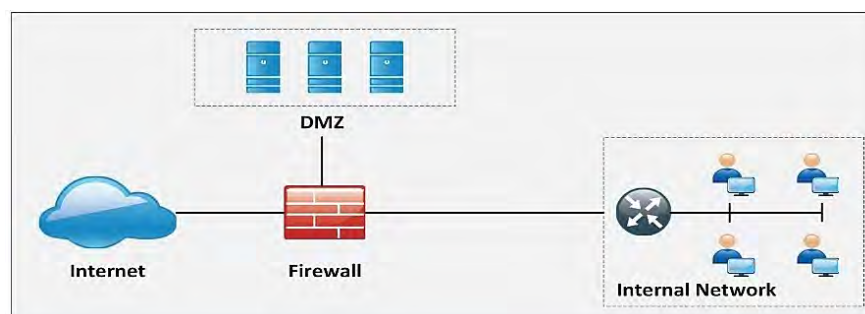


Figure 12-06: Screened Subnet

Multi-homed Firewall

A Multi-homed Firewall is two or more networks where each interface is connected to its network. It increases the efficiency and reliability of a network. A firewall with two or more interfaces allows further subdivision.

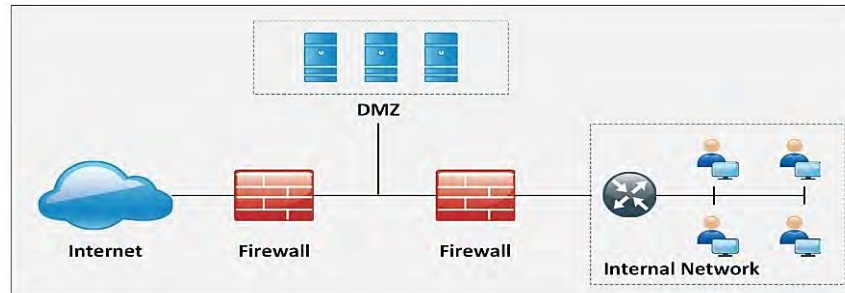


Figure 12-07: Multi-homed Firewall

Demilitarized Zone (DMZ)

An IOS zone-based firewall is a specific set of rules that may help to mitigate mid-level security attacks in environments where security is implemented via routers. In Zone-based Firewalls (ZBF), device interfaces are placed in different unique zones (inside, outside, or DMZ), and then policies are applied to these zones. Naming conventions for zones must be easy to understand in order to be helpful when it comes to troubleshooting.

ZBFs also use stateful filtering, which means that if the rule is defined to permit originating traffic from one zone to another zone, for example, DMZ, then return traffic is automatically allowed. Traffic from different zones can be allowed using policies permitting traffic in each direction.

One of the advantages of applying policies on zones rather than interfaces is that whenever new changes are required at the interface level, policies are applied automatically simply by removing or adding to an interface in a particular zone.

ZBF may use the following set of features in its implementation:

- Stateful Inspection
- Packet Filtering
- URL Filtering
- Transparent Firewall
- Virtual Routing Forwarding (VRF)

Figure 12-08 illustrates the scenario explained above.

Advantages	Disadvantages
Ease of implementation by using a permit and deny statements	Cannot mitigate IP spoofing attacks. An attacker can compromise the digital assets by spoofing the IP source address to one of the permit statements in the ACL
Less CPU intensive than deep packet inspection techniques	Difficult to maintain when ACL's size grows
Configurable on almost every Cisco IOS	Cannot implement filtering based on session states
Even a mid-range device can perform ACL-based filtering	In scenarios in which dynamic ports are used, a range of ports will be required to be opened in ACL, which malicious users may also use

Table 12-05: Advantages and Disadvantages of Packet Filtering Techniques

Circuit-level Gateway Firewall

A Circuit-level Gateway Firewall operates at the session layer of the OSI model. It captures the packet to monitor the TCP Handshake in order to validate whether the sessions are legitimate. Packets forwarded to the remote destination through a circuit-level firewall appear to be originated from the gateway.

Application-level Firewall

An application-level firewall can work at layer 3 up to layer 7 of the OSI model. Normally, specialized or open-source software running on a high-end server act as an intermediary between the client and destination address. As these firewalls can operate up to layer 7, it is possible to control moving in and out of more granular packets. Similarly, it becomes very difficult for an attacker to get the topology view of a trusted network because the connection request terminates on Application/Proxy firewalls.

Some of the advantages and disadvantages of using application/proxy firewalls are mentioned in Table 12-06.

Advantages	Disadvantages
Granular control over traffic is possible by using information up to layer 7 of the OSI model	As proxy and application, firewalls run in software. A very high-end machine may be required to fulfill the computational requirements
The indirect connection between end devices make it very difficult to generate an attack	Just like NAT, not every application has support for proxy firewalls, and few amendments may be needed in the current application architecture
Detailed logging is possible as every session involves the firewall as an intermediary	Other software may be required for the logging feature, which takes extra processing power
Any commercially available hardware can be used to install and run proxy firewalls on it	Along with computational power, high storage may be required in different scenarios

Table 12-06: Advantages and Disadvantages of Application/Proxy Firewalls

Stateful Multilayer Inspection Firewalls

As the name suggests, this saves the state of current sessions in a table known as a stateful database. Stateful inspection and firewalls using this technique normally deny any traffic between trusted and untrusted interfaces. Whenever an end device from a trusted interface wants to communicate with some destination address attached to the untrusted interface of the firewall, it will be entered into a stateful database table containing layer 3 and layer 2 information.

Advantages	Disadvantages
Helps in filtering unexpected traffic	Unable to mitigate application-layer attacks
Can be implemented on a broad range of routers and firewalls	Except for TCP, other protocols do not have well-defined state information to be used by the firewall
Can help in mitigating denial of service (DDoS) attacks	Some applications may use more than one port for a successful operation. An application architecture review may be needed in order to work after the deployment of the stateful inspection-based firewall.

Table 12-07: Advantages and Disadvantages of Stateful Inspection-based Firewalls

Application Proxy

An application-level proxy acts as a proxy server that filters connections for specific services. It allows traffic to pass through based on the services and protocols it supports. For instance, an FTP proxy will only permit FTP traffic while blocking all other services and protocols. This type of server serves as an interface between a user's workstation and the Internet, functioning alongside a gateway server to separate the enterprise network from the Internet.

When a user requests an internet service, the proxy receives the request and responds accordingly. A proxy service is essentially an application or program that forwards user requests (such as FTP or Telnet) to the actual services. It is also referred to as an application-level gateway because it establishes and renews connections to various services.

Proxies typically operate on a firewall host, which may be a dual-homed host or another secure bastion host. Certain types of proxies, known as caching proxies, enhance network efficiency by storing copies of requested data from the hosts they serve. These caching proxies can deliver the data directly when multiple hosts request the same information, thereby reducing the load on network connections.

While proxy servers provide both security and caching benefits, they operate transparently for the user. Instead of communicating directly with an external service, the user interacts with the proxy, which manages all communication between users and Internet services. The key advantage of proxy services is their transparency. To the user, it appears as if they are directly interacting with the real server, while to the server, the proxy presents the illusion that it is communicating directly with the user.

Network Address Translation (NAT)

Network Address Translation (NAT) divides IP addresses into two sets, allowing a Local Area Network (LAN) to use these addresses for both internal and external traffic. NAT helps conceal the layout of an internal network and forces all connections to pass through a central point. It operates in conjunction with a router and, similar to packet filtering, modifies the packets being sent through the router.

When an internal machine sends a packet to an external machine, NAT changes the source address to make it appear as if it is coming from a valid public address. Conversely, when the external machine responds, NAT modifies the destination address, converting the visible address back to the correct internal address. NAT can also adjust the source and destination port numbers.

By doing this, NAT limits the number of public IP addresses an organization needs. Additionally, it acts as a firewall filtering technique, permitting only those connections that originate within the internal network and blocking connections that come from the external network.

Virtual Private Networks (VPN) Firewalls

A Virtual Private Network (VPN) is a service that provides secure access to a private network over the Internet. VPNs connect wide area networks (WANs) and allow computers on one network to communicate with computers on another network. They are essential for securely transmitting sensitive information over untrusted networks through methods like encapsulation and encryption.

VPNs use encryption and integrity protection, which enable the use of a public network as if it were a private one. The VPN performs encryption and decryption outside the packet-filtering perimeter, allowing for the inspection of packets coming from different sites. It establishes a virtual point-to-

point connection using dedicated connections and encapsulates packets sent over the Internet. This combination leverages the benefits of both public and private networks.

While VPN technology is distinct from firewalls, firewalls can enhance VPN functionalities. They provide secure remote access services by encrypting data and managing traffic flow between VPN endpoints, ensuring that only authorized traffic passes through the VPN tunnel and implementing firewall rules to prevent unauthorized access. Additionally, a computing device running VPN software is restricted to accessing only the VPN.

All VPNs that run over the Internet adopt the following principles:

- Encrypts the traffic
- Checks for integrity protection
- Encapsulates new packets, which are sent across the Internet to some destination that reverses the encapsulation
- Checks the integrity
- ***Decrypts the traffic eventually***

Next Generation Firewalls (NGFW)

NGFW is used for the latest firewalls with advanced feature sets. This kind of firewall provides in-depth security features to mitigate known threats and malware attacks. An example of next-generation firewalls is the Cisco ASA series with FirePOWER services. NGFW provides complete visibility into network traffic users, mobile devices, Virtual Machines (VM) to VM data communication, etc.

Firewall Limitations

While firewalls are crucial to your security framework, they come with several limitations:

- Firewalls may block users from accessing useful services like FTP, Telnet, NIS, etc., and can occasionally limit Internet access as well
- A firewall is incapable of preventing internal threats (backdoors) in a network, such as an unhappy employee collaborating with an external attacker
- The focus of the firewall's security is concentrated at a single point, which makes other systems within the network vulnerable to attacks
- A congestion issue may arise if all connections are funneled through the firewall
- The firewall cannot shield the network from social engineering and data-driven threats where the attacker sends harmful links and emails to individuals inside the network
- If external devices like laptops, mobile phones, portable hard drives, etc., are already compromised and connected to the network, the firewall cannot safeguard the network from these devices
- The firewall is also unable to effectively defend against all varieties of zero-day viruses that seek to circumvent it
- A firewall is ineffective if the network's design and configuration are flawed

- A firewall should not be considered as a replacement for antivirus or antimalware software
- A firewall does not block attacks originating from higher levels of the protocol stack
- A firewall does not prevent assaults coming from common ports and applications
- A firewall fails to stop attacks from dial-in connections
- A firewall is not equipped to comprehend tunneled traffic

IDS, IPS, and Firewall Solutions

The previous section covered the functions, roles, and placements of Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), firewalls, and honeypots in network security. There are numerous user-friendly and feature-rich solutions, both hardware and software, available for implementing these security measures.

Intrusion Detection using YARA Rules

YARA is a tool for malware research that enables security analysts to identify and categorize malware or other harmful code using a rule-based methodology. It is a versatile tool compatible with Windows, macOS, and Linux operating systems. The tool allows security analysts to formulate "rules" or descriptions of malware families in textual or binary pattern form. The rules created analyze specific file patterns and notify security analysts if the file poses a threat. Each description or rule is made up of a Boolean expression and strings that define its underlying logic. Security analysts can also draft YARA rules to investigate a private database or malicious binaries within an organization to identify potential intrusions. In addition, they can utilize various patterns, such as hexadecimal or plaintext, along with additional special operators and strings in a YARA rule to effectively identify a broad range of malware signatures.

YARA rules follow a specific syntax, beginning with the term "rule" followed by its name. A rule consists of three components, which include:

1. **Condition:** This segment of a YARA rule specifies when the outcome will be true for a file that might be examined. It contains Boolean expressions that help define the matching criteria or result.
2. **Strings:** This provides substance to the "condition" section by delineating all the strings that should be searched for within the files. The rule can scan various types of strings, such as hexadecimal strings, text strings, or regular expressions.
3. **Metadata:** This element of a YARA rule includes general details that can aid a security analyst in recognizing the files identified by a specific rule.

An example of a YARA rule as per YARA's official documentation is as follows:

```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true
```

```

strings:
    $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
    $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
    $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"
condition:
    $a or $b or $c
}

```

From the above syntax, security experts can identify that any of those three strings defined must be considered as the u Trojan. In addition, the strings in the above example are comprised of both hex and text patterns.

YARA Tools

yarGen

yarGen is a tool designed for generating YARA rules. Its main function is to create YARA rules from strings found in malware files while eliminating any strings that also appear in clean (goodware) files. The tool comes with large ZIP archives containing goodware strings and an opcode database, which need to be extracted prior to use. Users can install all necessary dependencies by running the command **pip install -r requirements.txt**. Additionally, to obtain more information about command line parameters, users can execute **python yarGen.py --help**.

```

1  /*
2      Yara Rule Set
3      Author: YarGen Rule Generator
4      Date: 2022-07-09
5      Identifier: bin
6  */
7
8  /* Rule Set ----- */
9
10 rule backdoor {
11     meta:
12         description = "Auto-generated rule - file backdoor.exe"
13         author = "YarGen Rule Generator"
14         reference = "not set"
15         date = "2022-07-09"
16         hash = "bad8c7e6836b9a5679bfac0bc7483091e8e168f2"
17     strings:
18         $s0 = "%systemroot%\System32\rundll32.exe \" fullword ascii /* PESTudio Blacklist: str
19         $s1 = "c:\\Agenti\\SimpleVector\\Release\\SimpleVector.pdb" fullword ascii /* score: '28.
20         $s2 = "GetCurrentProcessID" fullword ascii /* PESTudio Blacklist: strings */ /* score: '2
21         $s3 = "<requestedExecutionLevel level=\"highestAvailable\" uiAccess=\"false\"/>" fullword
22         $s4 = "SOFTWARE\\Microsoft\\VisualStudio\\9.0\\Setup\\VS" fullword ascii /* PESTudio Blac
23         $s5 = "BG:\\oMpp" fullword ascii /* score: '12.00' */
24         $s6 = "vvKPP80k.mKn" fullword ascii /* score: '12.00' */
25         $s7 = "<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"no\" ?><assembly xmlns=\"u
26         $s8 = "0b55581e0a49451a01584c2a1d5223224559566318244a41405b172e11161932" fullword ascii /
27         $s9 = "SimpleVector, Version 1.0" fullword wide /* score: '9.00' */
28         $s10 = "* GN37" fullword ascii /* score: '7.00' */
29         $s11 = "SIMPLEVECTOR" fullword wide /* score: '6.50' */
30         $s12 = ".0cL/2" fullword ascii /* score: '6.00' */
31         $s13 = "VH.IYl" fullword ascii /* score: '6.00' */
32         $s14 = "uKmtnQzd78" fullword ascii /* score: '5.00' */
33         $s15 = "About SimpleVector" fullword wide /* score: '5.00' */
34     condition:
35         uint16(0) == 0x5a4d and filesize < 3785KB and all of them
36 }

```

Figure 12-09: Yara Rules creation using yarGen

Intrusion Detection Tools

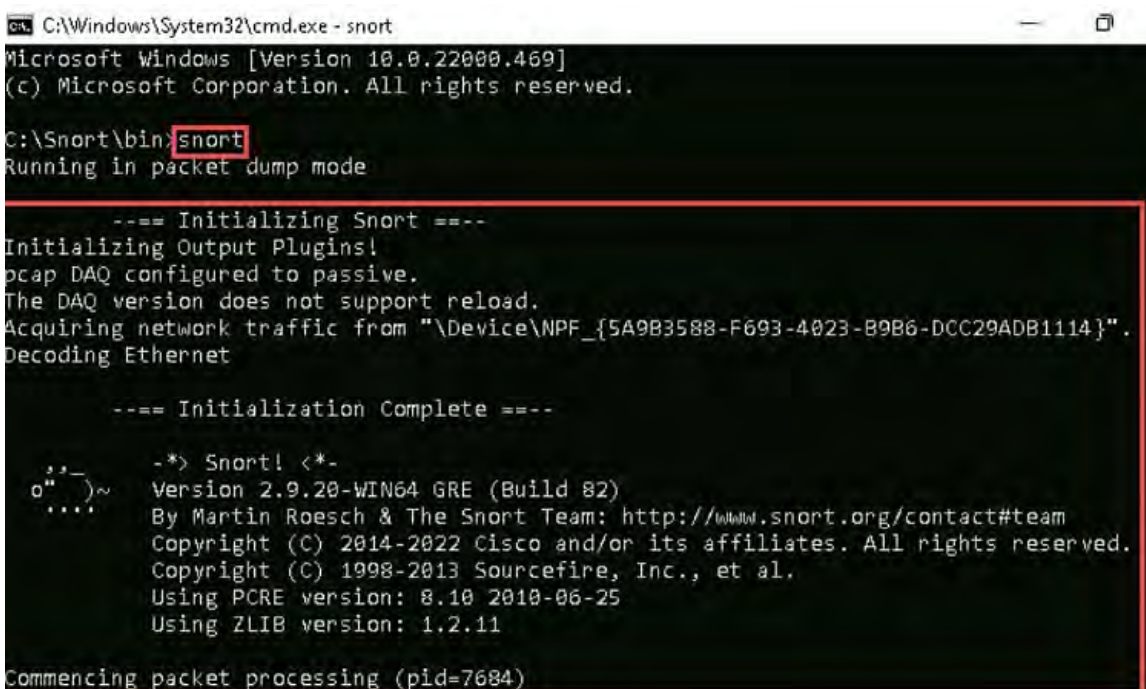
Intrusion detection tools identify irregularities. When operating on a dedicated workstation, these tools analyze all network packets, reconstruct user sessions, and search for potential intrusions by examining attack signatures and statistical anomalies in network traffic. Furthermore, these tools provide real-time protection against zero-day vulnerabilities from network attacks and harmful traffic, safeguarding against malware, spyware, port scans, and viruses, as well as DoS and DDoS attacks that could compromise hosts.

Snort

Snort is a network intrusion detection system that is open-source and can conduct real-time analysis of network traffic and log packets on IP networks. It is capable of protocol analysis and searching for/matching content. It is utilized to identify various types of attacks and probing activities, such as buffer overflow incidents, stealth port scanning, CGI attacks, SMB probing, and attempts at OS fingerprinting. It employs a versatile rules language to define the traffic it should capture or allow, along with a detection engine that incorporates a modular plug-in architecture.

Uses of Snort:

- Basic packet sniffer similar to tcpdump
- Packet logging tool (helpful for debugging network traffic, etc.)
- Network intrusion prevention system



```
C:\Windows\System32\cmd.exe - snort
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Snort\bin>snort
Running in packet dump mode

---- Initializing Snort ----
Initializing Output Plugins!
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "\Device\NPF_{5A9B3588-F693-4023-B9B6-DCC29ADB1114}".
Decoding Ethernet

---- Initialization Complete ----

o"~
...~
-*> Snort! <*-
Version 2.9.20-WIN64 GRE (Build 82)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using PCRE version: 8.10 2010-06-25
Using ZLIB version: 1.2.11

Commencing packet processing (pid=7684)
```

Figure 12-10: Snort Screenshot


```

C:\Windows\System32\cmd.exe - snort -dev -i 1
00 08 00 2F 80 01 00 00 00 78 00 06 C0 88 00 02 .../....X....
00 08 C0 AD 00 2F 80 01 00 00 00 78 00 06 C0 AD .../....X....
00 02 00 08 C0 7E 00 2F 80 01 00 00 00 78 00 08 ...~/.X..
C0 7E 00 04 40 00 00 08 ~.~.~.

+++++

WARNING: No preprocessors configured for policy 0.
03/12-22:20:03.807530 02:15:5D:20:CA:65 -> 33:33:00:00:00:FB type:0x0600 len:0x173
fe80:0000:0000:0000:0015:5dff:fe20:ca65:5353 -> ff02:0000:0000:0000:0000:0000:00fb:5353 UDP TTL:255 TOS:0x0 ID:0 Iplen:4
0 DgmLen:357
Len: 309
00 00 04 00 00 00 00 06 00 00 00 03 10 61 64 62 .....adb
20 75 6E 69 64 65 6E 74 69 66 69 65 64 04 5F 61 -unidentified..a
64 62 04 5F 74 63 70 05 0C 6F 63 61 6C 00 00 10 db_tcp.local...
00 01 00 00 11 94 00 01 00 09 5F 73 65 72 76 69 ....._servi
63 65 73 07 5F 64 6E 73 2D 73 64 04 5F 75 64 70 ces_dns-sd_udp
C0 27 00 0C 00 01 00 00 11 94 00 02 C0 1D C0 1D '.....
00 0C 00 01 00 00 11 94 00 02 C0 0C C0 0C 00 21 .....l
00 01 00 00 00 78 00 10 00 00 00 00 15 03 07 41 .....X.....A
5E 64 72 6F 69 64 C0 27 01 35 01 36 01 41 01 43 ndroid."5.6.A.C
01 30 01 32 01 45 01 46 01 46 01 46 01 44 01 35 .0.2.E.F.F.D.5
01 35 01 31 01 30 01 30 01 30 01 30 01 30 01 30 .5.1.0.0.0.0.0
01 30 01 30 01 30 01 30 01 30 01 30 01 30 01 30 .0.0.0.0.0.0.0
01 30 01 30 01 45 01 46 03 69 70 36 04 61 72 70 .0.8.E.F.Ip6.arp
01 00 00 0C 80 01 00 00 00 78 00 02 C0 7E C0 7E a.....X.....
00 1C 80 01 00 00 00 78 00 10 FE 80 00 00 00 00 .....X.....
00 00 00 15 5D FF FE 20 CA 65 C0 0C 00 2F 80 01 ....]..e.../..
00 00 11 94 00 09 C0 0C 00 05 00 00 80 00 40 C0 .....@.
00 00 2F 80 01 00 00 00 78 00 06 C0 88 00 02 00 .../....X.....
00 C0 7E 00 2F 80 01 00 00 00 78 00 08 C0 7E 00 ~.~.~.X.....
04 00 00 00 00 .....
  
```

Figure 12-11: Snort Output

Snort Rules

Snort's rule engine allows for the creation of custom rules tailored to network needs, differentiating between normal and malicious activities. It utilizes the **libpcap** library for packet sniffing, enabling monitoring of all packets through the network in promiscuous mode. Alerts are generated based on packet content and defined rules.

Users can write Snort rules to address security policy violations, common exploit attempts, and unusual packet conditions. These rules should be robust and flexible, providing hard checks on network activities while being adaptable enough to respond to intrusions.

When writing Snort rules, keep these principles in mind:

- Rules should be short, precise, and contained within a single line.
- Each rule consists of two sections: a rule header (detailing action, protocol, IP addresses, and ports) and rule options (including alert messages and inspected packet details).

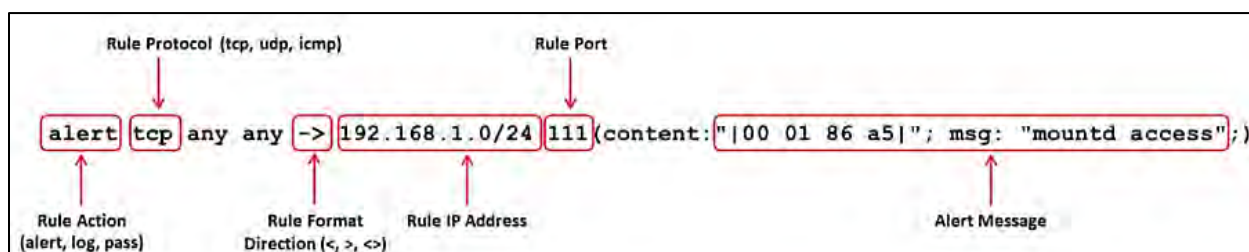


Figure 12-12: Snort Rules Example

Snort Rules: Rule Actions and IP Protocols

The rule header contains a complete set of rules for identifying packets and determining actions based on their attributes. It includes the rule action, which instructs Snort on what to do when a matching packet is found. Snort supports six default actions: alert, log, pass, drop, reject, and sdrop.

Data is sent over the Internet using the IP, which provides unique addressing for each computer and organizes data into packets containing message data, source, and destination. Snort addresses suspicious behavior using three IP protocols:

- **Transmission Control Protocol (TCP):** Used for connecting two hosts and exchanging data.
- **User Datagram Protocol (UDP):** Used for broadcasting messages.
- **Internet Control Message Protocol (ICMP):** Used by the OS to send error messages.

Snort Rules: Direction Operator and IP Addresses

Direction Operator

This operator signifies the preferred direction of traffic; it can move either in one direction or in both directions. An example of a Snort rule that employs the Bidirectional Operator is:

```
log tcp !192.168.1.0/24 any <> 192.168.1.0/24 23
```

IP Addresses

- Specify the IP address and port that the rule targets
- Use the keyword "any" to define the IP address
- Employ numeric IP addresses with a CIDR netmask

Example of an IP address negation rule:

```
alert tcp !192.168.1.0/24 any -> 192.168.1.0/24 i!! (content: "|00 01 86 a5|"; msg: "external mountd access");
```

Snort Rules: Port Numbers

Port numbers can be specified in various ways, such as "any" ports, static definitions, port ranges, and negation. Port ranges use the ":" operator. The direction operator "->" indicates traffic orientation, with the left side representing the source and the right side representing the destination. A bidirectional operator "<>" allows Snort to consider both source and destination pairs, which is useful for sessions like telnet or POP3. There is no "<-" operator. Snort rules also define source and destination IP addresses and ports, allowing for single or multiple addresses separated by commas and enclosed in square brackets, such as:

```
[192.168.1.1,192.168.1.45,10.1.1.24]
```

Ensure there are no whitespaces when specifying IP address ranges using CIDR notation. Snort supports the NOT operator ("!") to exclude specific IPs or CIDR ranges from rules. For example, you can create a rule to alert traffic from outside the local network using the negation operator.

For Example:

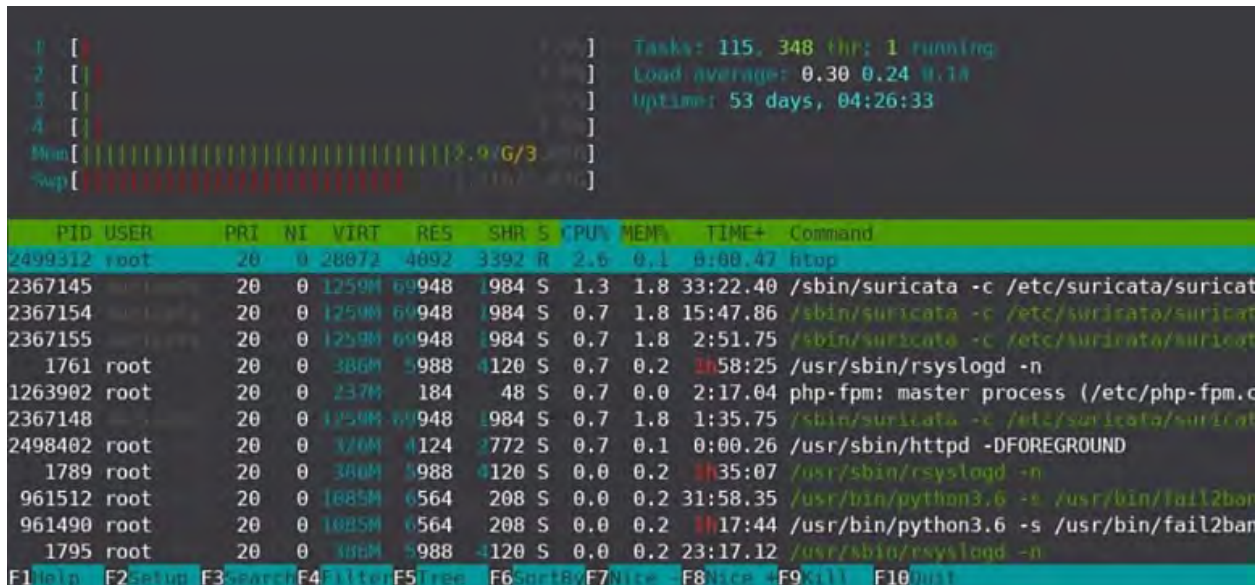
```
log tcp any -> 192.168.1.0/24 !6000:6010
```

Protocols	IP Address	Action
Log UDP any ->	192.168.1.0/24 !:1024	Log UDP traffic coming from any port and destination ports ranging from 1 to 1024
Log TCP any ->	192.168.1.0/24 :5000	Log TCP traffic from any port going to ports less than or equal to 5000
Log TCP any :1024 ->	192.168.1.0/24 400:	Log TCP traffic from the well-known ports and going to ports greater than or equal to 400

Table 12-08: Examples of a Port Negation

Suricata

Suricata is a powerful engine designed for detecting network threats, offering capabilities for inline intrusion prevention (IPS), real-time intrusion detection (IDS), Network Security Monitoring (NSM), and offline pcap analysis. It examines network traffic using extensive rules and a sophisticated signature language, along with strong support for Lua scripting to identify complex threats. Using standard input and output formats like YAML and JSON, it seamlessly connects with existing tools like SIEMs, Splunk, Logstash/Elasticsearch, Kibana, and various other databases.



```

0 [ ] Tasks: 115, 348 thr: 1 running
1 [ ] Load average: 0.30 0.24 0.18
2 [ ] Uptime: 53 days, 04:26:33
3 [ ]
4 [ ]
Mem[ ] 2.9G/3.0G
Swap[ ] 0.0G/0.0G

```

PID	USER	PRI	NI	VRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
2499312	root	20	0	28872	4892	3392	R	2.6	0.1	0:00.47	htop
2367145	suricata	20	0	1259M	60948	1984	S	1.3	1.8	33:22.40	/sbin/suricata -c /etc/suricata/suricat
2367154	suricata	20	0	1259M	60948	1984	S	0.7	1.8	15:47.86	/sbin/suricata -c /etc/suricata/suricat
2367155	suricata	20	0	1259M	60948	1984	S	0.7	1.8	2:51.75	/sbin/suricata -c /etc/suricata/suricat
1761	root	20	0	386M	5988	4120	S	0.7	0.2	36:58:25	/usr/sbin/rsyslogd -n
1263902	root	20	0	237M	184	48	S	0.7	0.0	2:17.04	php-fpm: master process (/etc/php-fpm.c
2367148	suricata	20	0	1259M	60948	1984	S	0.7	1.8	1:35.75	/sbin/suricata -c /etc/suricata/suricat
2498402	root	20	0	326M	4124	2772	S	0.7	0.1	0:00.26	/usr/sbin/httpd -DFOREGROUND
1789	root	20	0	386M	5988	4120	S	0.0	0.2	36:35:07	/usr/sbin/rsyslogd -n
961512	root	20	0	1085M	6564	208	S	0.0	0.2	31:58.35	/usr/bin/python3.6 -s /usr/bin/fail2ban
961490	root	20	0	1085M	6564	208	S	0.0	0.2	36:17:44	/usr/bin/python3.6 -s /usr/bin/fail2ban
1795	root	20	0	386M	5988	4120	S	0.0	0.2	23:17.12	/usr/sbin/rsyslogd -n

```

F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 Sort By F7 Nice F8 Sica F9 Kill F10 Quit

```

Figure 12-13: Screenshot of Suricata

Intrusion Prevention Tools

Trellix Intrusion Prevention System

The Trellix Intrusion Prevention System assists security experts in identifying covert botnets, worms, and reconnaissance attacks early on. It consolidates flow data from routers and switches, conducting network-level threat behavior analysis to identify unusual patterns of behavior. It detects and prevents sophisticated threats on-site within virtual environments, software-defined data centers, and both private and public clouds.

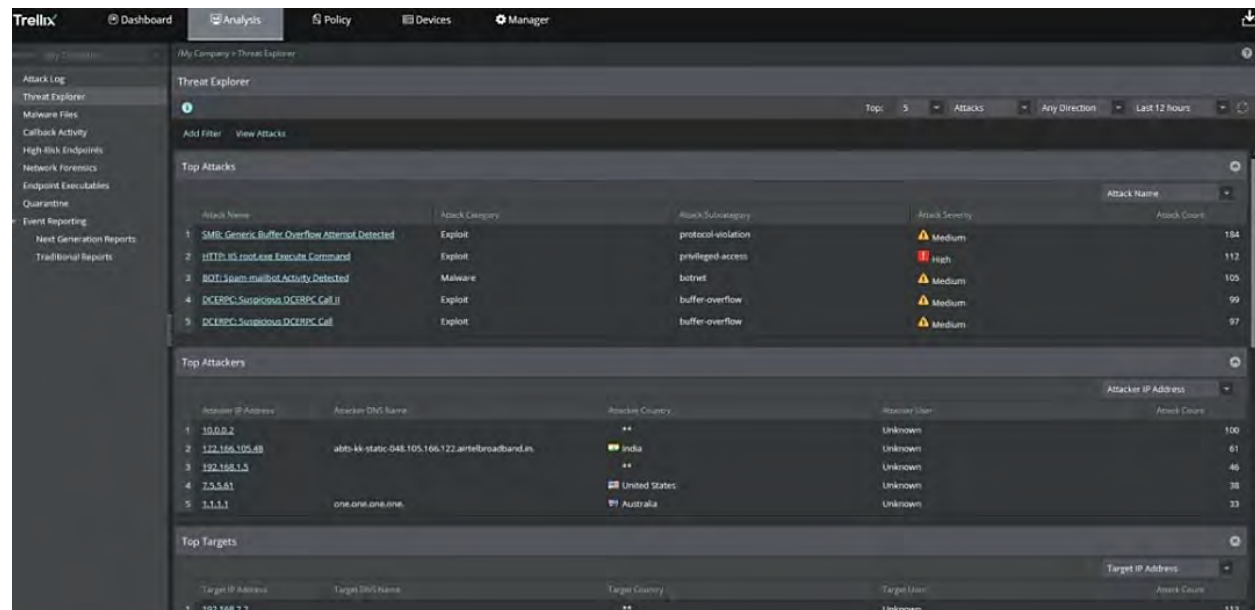


Figure 12-14: Screenshot of Trellix Intrusion Prevention System

Firewalls

Firewalls play a crucial role in protecting computers from viruses, privacy threats, inappropriate content, hackers, and malicious software while connected to the Internet. They monitor the applications running on a device that accesses the network. Firewalls analyze downloads, raise alerts when a potentially harmful file is detected, and block it to prevent infection of the computer.

Firewalls Based on Configuration

Network-based Firewalls

- Cisco Secure Firewall ASA
- PA-7500
- FortiGate 7121F
- Check Point 28000 Quantum Security Gateway
- Juniper SRX

Host-based Firewalls

- Microsoft Defender Firewall
- ZoneAlarm Pro Firewall
- Comodo Firewall

- Norton Smart Firewall
- McAfee Firewall

Firewalls Based on Working Mechanism

- IPFire
- pfSense
- SonicWall TZ Series
- BIG-IP Advanced Firewall Manager
- Sophos Firewall
- WatchGuard Firebox
- FortiProxy
- SonicWall NSa 6700
- ZYXEL VPN Firewall
- DrayTek Vigor2765

Evading IDS/Firewalls

Avoiding Intrusion Detection Systems (IDS) and firewalls is an essential strategy for attackers aiming to bypass security protocols and gain unauthorized entry into networks. This enables attackers to remain undetected while exploiting weaknesses, thus extending their time within a network without triggering alerts. Techniques such as payload obfuscation, encrypted communication channels, and the use of zero-day vulnerabilities are frequently utilized to evade these protective measures. Comprehending these evasion techniques is vital for security experts, as it helps them enhance their defenses, create more robust IDS signatures, and configure firewalls to better identify and prevent advanced attack patterns.

IDS/Firewall Evasion Techniques

Evading an IDS/firewall refers to a strategy where an attacker modifies the sequence of an attack to escape detection by the security system in place. An IDS/firewall functions based on a specific set of rules, and with adequate knowledge and expertise, an attacker can navigate around these protections by utilizing a variety of methods. By implementing these techniques, the attacker is able to deceive a security system without filtering the harmful traffic they create.

Several techniques for bypassing IDS/firewalls include the following:

- Port Scanning
- Firewalking
- Banner Grabbing
- IP Address Spoofing
- Source Routing
- Tiny Fragments
- Using an IP Address in Place of a URL
- Using Anonymous Website Surfing Sites

- Using a Proxy Server
- ICMP Tunneling
- ACK Tunneling
- HTTP Tunneling
- SSH Tunneling
- DNS Tunneling
- Through External Systems
- Through MITM Attack
- Through Content
- Through XSS Attack
- Through HTML Smuggling
- Through Windows BITS

Port Scanning

Port Scanning is an examination procedure mostly used by attackers to identify the open port. However, legitimate users may also use it. Port scanning does not always lead to an attack, as the user and attacker use it. It can be used to collect information before an attack. In this scenario, special packets are forwarded to a particular host whose response is examined by the attacker to get information regarding open ports.

Firewalking

Firewalking is a technique in which an attacker, using an ICMP packet, finds out the location of the firewall and networking map by probing the ICMP echo request with TTL values incrementing one by one. It helps the attacker to find out the number of hops.

Banner Grabbing

Banners are service announcements generated by services in response to connection requests, and they frequently include vendor version information. Banner grabbing is a simple fingerprinting technique that aids in identifying a firewall's firmware version and provider. Understanding the precise firmware or server version can aid in the development of payloads that take advantage of known vulnerabilities and evade detection by the IDS patterns. The system's services can be identified via banner grabbing. Details on the target systems' setups or IDS are among the information gleaned by banner capturing. With this information, attackers can comprehend how an intrusion detection system is configured and modify its operations to circumvent established detection guidelines. Banner snatching is another method that attackers might use to find out which services are operating on firewalls. Web servers, Telnet, and FTP are the three main services that send out banners.

IP Address Spoofing

IP Address Spoofing is a technique used to gain unauthorized access to machines by spoofing the IP address. An attacker illicitly impersonates any user machine by sending manipulated IP packets

with a spoofed IP address. The spoofing process involves modifying the header with a spoofed source IP address, a checksum, and the order values.

Source Routing

Source Routing is the technique of sending a packet via a selected route. In session hijacking, this technique is used to attempt IP spoofing as a legitimate host, and with the help of source routing, the traffic is directed through a path identical to the victim's path.

Tiny Fragments

Attackers create tiny fragments of outgoing packets that disrupt the TCP header information by forcing it into subsequent fragments. This fragmentation complicates the ability of Intrusion Detection Systems (IDS) to reassemble the traffic stream, hampering the detection of malicious patterns. If a filtering router examines only the first fragment and allows the others through, the attack can succeed by bypassing user-defined filtering rules, especially when the firewall checks only the TCP header.

Bypass Blocked Sites Using an IP Address in Place of a URL

In this technique, a blocked website in a network is accessed using the IP address. Consider a firewall blocking the incoming traffic destined for a particular domain. It can be accessed by typing the IP address in the URL instead of the domain name unless the IP address is also configured in the access control list.

Bypass Blocked Sites Using Anonymous Website Surfing Sites

Anonymous web-surfing sites allow you to browse the internet without revealing your identity and help you access blocked sites, effectively bypassing firewall restrictions. By using these sites, you can visit restricted websites without exposing your IP address. There are various anonymous web-surfing options available, some of which also offer the option to encrypt website URLs for added security.

Bypass an IDS/Firewall Using a Proxy Server

Accessing a blocked website using a proxy is very common. There are many online proxy solutions available that can hide your actual IP address and allow access to restricted websites.

Bypassing an IDS/Firewall through the ICMP Tunneling Method

ICMP tunneling is a technique of injecting arbitrary data into the payload of an echo packet and forwarding it to the targeted host. ICMP tunnels function on ICMP echo requests and reply packets. Using ICMP tunneling, TCP communication is tunneled over a ping request. A reply is received because most firewalls do not examine the payload field of the ICMP packets. Also, some network administrators allow ICMP for troubleshooting purposes.

Bypassing an IDS/Firewall through the ACK Tunneling method

ACK tunneling is a technique used by attackers to evade firewalls and IDS by exploiting the trusted nature of ACK packets. Firewalls typically focus on filtering SYN packets during TCP connection establishment, assuming malicious code is likely present in them. Once a session is active, ACK

packets, which confirm data receipt, are considered legitimate and are often filtered less rigorously. This oversight allows attackers to inject malicious payloads into ACK packets and tunnel backdoor applications, bypassing firewalls and potentially evading IDS detection.

Bypassing an IDS/Firewall through the HTTP Tunneling Method

HTTP Tunneling is another way of bypassing firewalls. Consider a company with a web server listening to traffic on port 80 for HTTP traffic. HTTP tunneling allows the attacker to evade the system despite the restriction imposed by the firewall encapsulating the data in the HTTP traffic. The firewall will allow port 80; an attacker may perform various tasks by hiding in the HTTP, for example, using FTP via HTTP protocol.

HTTP Tunneling Tools

- HTTP Port
- HTTPHost
- Super Network Tunnel
- HTTP-Tunnel

Bypassing Firewalls through the SSH Tunneling Method

OpenSSH is an encryption protocol for securing traffic from threats and attacks such as eavesdropping, hijacking, etc. An SSH connection is mostly used by applications to connect to application servers. An attacker uses OpenSSH to encrypt traffic to avoid detection by security devices.

Bypassing Firewalls through the DNS Tunneling Method

DNS tunneling is a technique used to bypass firewalls and intrusion detection systems by leveraging the trusted nature of DNS traffic. Operating over UDP, DNS queries are limited to 255 bytes and only allow alphanumeric characters and hyphens, making them suitable for covert communication. Attackers embed harmful data within DNS packets, splitting them into chunks for encoding in queries and responses, which allows data exfiltration without detection. DNSSEC cannot identify abnormalities in this process since the data is hidden within legitimate DNS traffic. Tools like iodine and dnscat2 are often employed to create and manage DNS tunnels.

Bypassing an IDS/Firewall through External Systems

Bypassing through an external system is the process of hijacking a legitimate user's session on a corporate network connected to an external network. An attacker can easily sniff the traffic to extract information, steal session IDs and cookies, and impersonate the user to bypass the firewall. An attacker can also infect the legitimate user using the external system with malware or Trojans to steal information.

Bypassing an IDS/Firewall through MITM Attacks

Many security administrators concentrate on the risk of external or internal networks bypassing their firewalls or Intrusion Detection Systems (IDS) while often overlooking the fact that these defenses can also be bypassed through Man-In-The-Middle (MITM) attacks on DNS servers. In MITM attacks, attackers exploit DNS servers and routing methods to bypass firewall and IDS

restrictions. They may either compromise the corporate DNS server or spoof DNS responses to execute the MITM attack.

Bypassing an IDS/Firewall through Content

In this method, attackers send files with malicious code to users, tricking them into opening these files to execute the code. They often use steganography or obfuscation to hide the harmful content within legitimate files, evading IDS and firewalls. For example, attackers may send emails with malicious executable files or documents that exploit macros. They can also target WWW/FTP servers, embedding Trojan horse files in software installations. Various file formats for text, multimedia, and graphics can carry this malicious content.

Commonly used file formats for carrying malicious content are:

- EXE, COM, BAT, PS, PDF CDR (Corel Draw)
- DVB, DWG (AutoCAD)
- SMM (AMI Pro)
- DOC, DOT, CNV, ASD (MS Word)
- XLS, XLB, XLT (MS Excel)
- ADP, MDA, MDB, MDE, MDN, MDZ (MS Access)
- VSD (Visio)
- MPP, MPT (MS Project)
- PPT, PPS, POT (MS PowerPoint)
- MSG, OTM (MS Outlook)

Bypassing an IDS/WAF using an XSS Attack

An XSS attack takes advantage of vulnerabilities that arise when a web application processes user input parameters and server responses. Attackers exploit these weaknesses to inject malicious HTML code into the target website, allowing them to bypass Intrusion Detection Systems (IDS) and Web Application Firewalls (WAF).

Techniques include:

- Using ASCII values
- Using Hex Encoding
- Using Obfuscation

Bypassing an IDS/Firewall through HTML Smuggling

HTML smuggling refers to a type of web attack where an attacker embeds harmful code within an HTML script to compromise a web page. This method enables attackers to manipulate the functionalities of scripting languages (such as HTML5 and JavaScript) while remaining undetectable by SIEM systems, firewalls, web proxies, and email filters. The goal of HTML smuggling is to effectively install a malware payload on a target system when the victim clicks a malicious link sent through a phishing email. An attacker crafts a malicious link by generating a JavaScript-based blob with a compatible MIME type that automatically triggers the download of the malware.

Tools like HTML Smuggler can be utilized to create a JavaScript payload that can bypass firewalls/IDS and deliver it through an HTML attachment. Once the malware runs, it grants the attacker remote access to carry out ongoing attacks.

Evading an IDS/Firewall through Windows BITS

In a Windows environment, the Background Intelligent Transfer Service (BITS) facilitates the distribution of automatic updates. While beneficial, BITS can be exploited by attackers to bypass firewalls and IDS systems, as organizations often overlook its traffic. It allows browsers like Firefox and Chrome to update even when closed. Attackers may use BITS to download malware or establish backdoors, enabling them to evade security measures and control systems. This occurs through BITS tasks that allow stealthy file transfers.

Other Techniques for Bypassing WAF

Using HTTP Header Spoofing

Web application firewalls enable certain queries and syntaxes that come from internal addresses. They also facilitate quick debugging of applications in testing environments. Malicious users exploit this feature by sending requests with forged headers, deceiving the targeted WAF and server into thinking that the request came from their internal network.



EXAM TIP: Attackers use tools such as Burp Suite to exploit HTTP headers and bypass WAF.

Using Blacklist Detection

Attackers may exploit the detection system of the WAF blacklist to bypass it. To achieve this, they analyze the specific WAF in order to discover the keywords that are blacklisted. By pinpointing these blacklisted keywords, attackers generate new regular expressions and payloads that do not contain any of the keywords on the blacklist.

Using Fuzzing/Brute-forcing

Attackers test a target WAF with known payloads to evade detection. They initially send payloads to a local WAF to identify effective evasion strategies. Using wordlists like Assetnote and SecLists, they perform fuzzing on the target network. They load the wordlist into a fuzzer, record responses, and utilize random user agents and proxy chains to avoid WAF detection.

Abusing SSL/TLS Ciphers

Attackers can exploit a vulnerability where a web server supports SSL/TLS ciphers that a filtering firewall (WAF) does not. They first analyze the WAF to identify its supported ciphers, often using vendor documentation and tools like sslscan2 (<https://github.com>). By finding a cipher supported by the web server but not the WAF, they can bypass the WAF using tools such as abuse-ssl-bypass-waf.py and curl to establish a connection with the target server.

Other Techniques for IDS Evasion

Intrusion Detection Systems (IDS) that enhance the security of an organization's infrastructure are appealing targets for attackers. Attackers utilize various techniques to evade IDS and compromise

the system's defenses. IDS evasion involves altering attack methods to deceive the IDS/IPS into perceiving the traffic as legitimate, thereby avoiding triggering an alert. Numerous IDS evasion techniques can effectively bypass detection in various ways.

Insertion Attack

An Insertion Attack is a kind of evasion of an IDS device done by taking advantage of users' blind belief in IDS. The Intrusion Detection System (IDS) assumes that the end systems also accept accepted packets, but there may be a possibility that the end system rejects these packets. This type of attack particularly targets Signature-based IDS devices to insert data into the IDS. Taking advantage of a vulnerability, an attacker can insert packets with bad checksum or TTL values and send them out of order. When reassembling the packet, the IDS and end host might have two different streams.

For example, figure 12-15 shows the attacker send packets whose time-to-live (TTL) fields are crafted to reach the IDS but not the target computers. This will result in the IDS and the target system having two different character strings. An attacker confronts the IDS with a stream of one-character packets (the attacker-originated data stream), in which one of the characters (the letter "X") will be accepted only by the IDS. As a result, the IDS and the end system reconstruct two different strings.

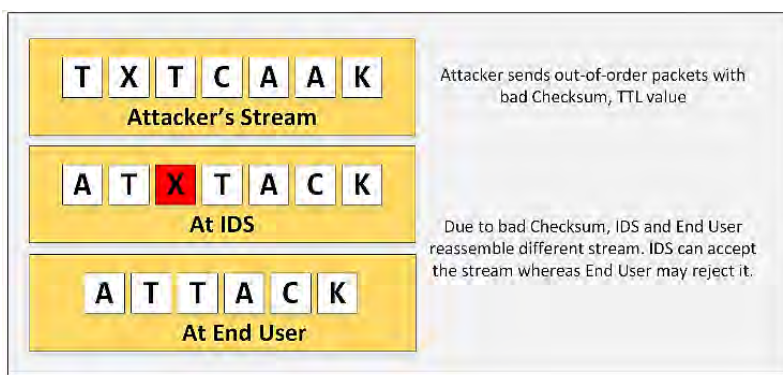


Figure 12-15: Insertion Attack on IDS

Evasion

Evasion is a technique intended to send a packet that is accepted by the end system, but the IDS rejects that. Evasion techniques are intended to exploit the host. An IDS that mistakenly rejects such a packet misses its contents entirely. An attacker may take advantage of this condition and exploit it.

For example, figure 12-16 shows an evasion attack, when an attacker opens a TCP connection with a data packet. Before any TCP connection can be used, it must be "opened" with a handshake between the two endpoints of the connection. An essential fact about TCP is that the handshake packets can themselves bear data. The IDS that does not accept the data in these packets is vulnerable to an evasion attack.

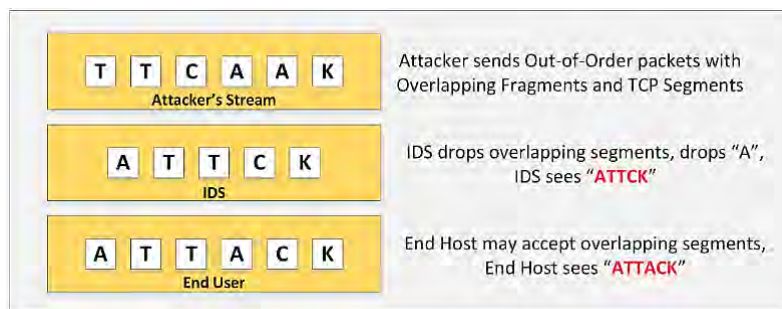


Figure 12-16: IDS Evasion

Fragmentation Attack

Fragmentation is the process of splitting a packet into fragments. This technique is usually adopted when the IDS and host device are configured with different timeouts. For example, if IDS is configured with 10 seconds of timeout while the host is configured with 20 seconds of timeout, sending packets with a 15-second delay will bypass reassembly at IDS and reassemble at the host.

Similarly, overlapping fragments can be sent. In overlapping fragmentation, a packet with the TCP sequence number configured is overlapping. Reassembly of these overlapping, fragmented packets depend on the Operating System. The host OS may use original fragmentation, whereas IOS devices may use subsequent fragments using offsets.



EXAM TIP: A simple way of splitting packets is by fragmenting them, but an adversary can also craft packets with small payloads. The whisker tool calls crafting packets with small payloads 'session splicing'. By itself, small packets will not evade any IDS that reassembles packet streams.

Denial-of-Service Attack (DoS)

Passive IDS devices are inherently Fail-Open rather than Fail-Closed. Taking advantage of this limitation, an attacker may launch a denial-of-service attack on the network to overload the IDS System. To perform a DoS attack on IDS, an attacker may target CPU exhaustion or Memory Exhaustion techniques to overload the IDS. These can be done by sending specially crafted packets that consume more CPU resources or sending a large number of fragmented out-of-order packets.

Obfuscating

Obfuscation is the encryption of a packet's payload destined to a target in such a way that the target host can reverse it, but the IDS cannot. It exploits the end-user without alerting the IDS, using different techniques such as encoding, encryption, and polymorphism. The IDS do not inspect encrypted protocols unless it is configured with the private key used by the server to encrypt the packets. Similarly, an attacker may use polymorphic shellcode to create unique patterns to evade IDS.

False Positive Generation

False Positive Alert Generation is the false indication of a result inspected for a particular condition or policy. An attacker may generate a large number of false-positive alerts by sending a suspicious packet containing real malicious packets to pass the IDS.

Session Splicing

Session Splicing is a technique in which an attacker splits the traffic into a large number of smaller packets in a way that not even a single packet triggers the alert. This can also be done by a slightly different technique, such as adding a delay between packets. This technique is effective for those IDS that do not reassemble the sequence to check against intrusion.

Unicode Evasion Technique

The Unicode Evasion Technique is another technique in which an attacker may use Unicode to manipulate the IDS. Unicode is a character encoding, as defined earlier in the HTML Encoding section. Converting strings using Unicode characters can prevent signature matching and alerting the IDS, thus bypassing the detection system.

Time-To-Live Attacks

Each IP packet contains a Time to Live (TTL) field, which indicates the number of hops a packet can take before being discarded. Routers decrement this value by 1, and when TTL reaches 0, the packet is dropped, sending an ICMP alert to the sender. Hosts typically set a high initial TTL, but different operating systems use varying default values. Attackers can estimate the number of routers and infer the host's OS based on the TTL value, which assists in preparing their attacks. To disrupt this detection, tools like SmartDefense can modify the TTL of outgoing packets. Attackers may gather network topology information using tools like traceroute.

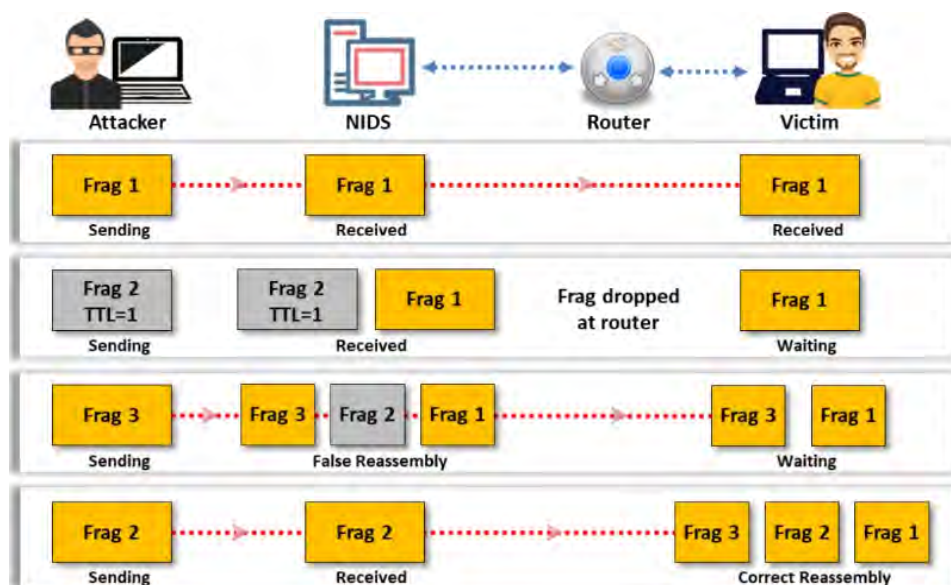


Figure 12-17: Evading IDS using Time-To-Live Attack

Urgency Flag

The urgency flag in TCP marks certain data as urgent, using an urgency pointer to indicate the start of this urgent data within a packet. When set, TCP ignores all data before the pointer and processes only the urgent data. Some Intrusion Detection Systems (IDS) do not account for this urgency feature and analyze all packets, while the target system only processes the urgent data. Attackers exploit this by placing irrelevant data before the urgent data, leading the IDS to see more data than what the target system processes. This discrepancy can be exploited to bypass IDS and transmit attack traffic.

Invalid RST Packets

TCP uses 16-bit checksums for error checking of headers and data, ensuring reliable communication. Each transmitted segment includes a checksum that the receiving end verifies; if there is a mismatch, the segment is dropped. TCP can also send RST packets to terminate communications, which attackers might exploit by sending RST packets with invalid checksums. This can mislead Intrusion Detection Systems (IDS) into thinking the session has ended, allowing attackers to continue communicating with the host while the IDS fails to reassemble the stream. The end host will drop any invalid packets but may still process subsequent valid packets.

Polymorphic Shellcode

A signature-based Network Intrusion Detection System (NIDS) identifies attacks by comparing data packets with known attack signatures. However, polymorphic shellcode attacks complicate detection as they employ multiple signatures and encode payloads with various techniques, placing a decoder in front. This rewriting allows the shellcode to evade detection. Attackers typically exploit buffer overflow vulnerabilities, manipulating the stack to redirect the return address to the decryption code after an overflow, further masking the attack. Consequently, NIDS often fails to recognize modified attacks, making common shellcode signatures ineffective.

ASCII Shellcode

ASCII shellcodes consist solely of characters from the ASCII standard. Such shellcodes enable attackers to navigate around typical character restrictions found in string input code. They also assist attackers in evading IDS pattern-matching signatures by concealing strings in a manner akin to polymorphic shellcodes. The IDS pattern matching system is not very effective with ASCII values.

Utilizing ASCII for shellcode is quite limiting, as it restricts what the shellcode can accomplish in certain situations, given that not every assembly instruction can be directly represented in ASCII values. This limitation prevents the use of alternative instructions or a combination of instructions that would convert to ASCII character representations, which would serve the same function as those instructions that improperly convert.

Application-Layer Attacks

Media files like images, audio, and videos are often compressed for faster transfer. However, this compression can hide flaws that attackers exploit, making it difficult for Intrusion Detection Systems (IDS) to detect such malicious content. While many applications use compression to

enhance speed, vulnerabilities can be embedded within the compressed data, evading signature-based detection. Attackers can leverage various methods, such as exploiting integer overflow vulnerabilities with different values, complicating signature detection even further.

Desynchronization

Pre-Connection SYN

The attack involves sending an initial SYN packet before the actual connection is established but with an invalid TCP checksum. The Intrusion Detection System (IDS) may either ignore or accept subsequent SYN packets in a connection. If a SYN packet is received after the TCP control block is opened, the IDS adjusts the appropriate sequence number to match the new SYN packet.

Attackers send fake SYN packets with completely invalid sequence numbers to desynchronize the IDS. This desynchronization prevents the IDS from effectively monitoring both legitimate and malicious traffic. If the IDS is designed intelligently, it will not check the TCP checksum. However, if the IDS does check the checksum, the attack becomes synchronized, as a bogus sequence number is sent to the IDS before the actual connection is established.

Post-Connection SYN In this technique, attackers aim to desynchronize the Intrusion Detection System (IDS) from the actual sequence numbers that the kernel is tracking. They do this by sending a post-connection SYN packet within the data stream. This packet will have different sequence numbers but will still meet all the necessary criteria for acceptance by the target host. However, the target host will ignore this SYN packet since it references an already established connection.

The goal of this attack is to cause the IDS to resynchronize its understanding of the sequence numbers to the new SYN packet. As a result, it will overlook any data that is legitimately part of the original stream, as it will be expecting a different sequence number. Once the attacker successfully resynchronizes the IDS with the SYN packet, they can then send a Reset (RST) packet using the new sequence number, effectively terminating the IDS's understanding of the connection.

Domain Generation Algorithms (DGA)

A domain generation algorithm (DGA) is a software program that attackers can employ to generate numerous new domain names and execute malware code. This program enables them to evade traditional detection mechanisms such as security gateways and signature filters, which generally block static IP addresses and specific domain names. This technique also helps them change domains frequently and dynamically identify a destination domain for command and control (C2) traffic, rather than relying on static IP addresses or domains.

DGAs use character sequences to generate a large number of random domain names, which serve as assembly points for attackers to communicate with the C2 servers. This method involves creating “gibberish” strings, for example, 'isdfcbjdjdnfuyt.ru,' while constructing the domain names by generating each letter. Attackers can use these newly generated, unregistered domain names to evade detection mechanisms that rely on IP reputation and the signatures of registered domains.

Encryption

Intrusion detection based on network analysis examines the data flow across the network from the origin to the endpoint. Suppose an attacker manages to create an encrypted connection with their target system through a Secure Shell (SSH), Secure Socket Layer (SSL), or a Virtual Private Network (VPN) tunnel. In that case, the Intrusion Detection System (IDS) will not inspect the data packets transmitted within these encrypted channels. As a result, an attacker can transmit harmful traffic through such secure connections, thus circumventing IDS protections.

Flooding

IDS utilizes memory and processing speed to scrutinize the incoming traffic. Attackers can bypass IDS security by overwhelming its resources with misleading or fake traffic, which causes the system to become exhausted from having to analyze the saturating traffic. When these types of attacks are successful, the attackers can direct harmful traffic to the target system located behind the IDS, which may provide minimal or no interference. As a result, the actual attack traffic may remain unnoticed.

Evading NAC and Endpoint Security

Evading Network Access Control (NAC) and endpoint security involves bypassing mechanisms that enforce security policies to gain unauthorized access to a network or system. NAC ensures that devices comply with security standards before granting them access, while endpoint security solutions detect and block potential threats.

NAC and Endpoint Security Evasion Techniques

NAC serves as a security measure designed to prevent unauthorized or unidentified devices/hosts from gaining access to internal services. Attackers frequently attempt to circumvent NACs using various methods to carry out malicious actions within the target network.

Some methods for bypassing NAC include:

- VLAN hopping
- Utilizing a pre-authenticated device

Endpoint security adds a layer of defense for end-user devices such as desktops, laptops, tablets, and digital printers against malware and other cyber threats. Nevertheless, attackers can utilize different evasion methods to sidestep Endpoint Detection and Response (EDR), allowing them to infect devices with potential malware and establish command and control while avoiding detection.

Some techniques for bypassing endpoint security include:

- VLAN Hopping
- Using Pre-authenticated Device
- Ghostwriting
- Using application whitelisting
- Dechaining macros

- Clearing memory hooks
- Process injection
- Using LoLBins
- Control Panel (CPL) side-loading
- Using ChatGPT
- Using Metasploit templates
- Windows Antimalware Scan Interface (AMSI)
- Hosting phishing sites
- Passing encoded commands
- Fast flux DNS method
- Timing-based evasion
- Signed binary proxy execution
- Shellcode encryption
- Reducing entropy
- Escaping the (local) AV sandbox
- Disabling event tracing for Windows
- Evading “mark of the Syscall”
- Spoofing the thread call stack
- In-memory encryption of beacon

Bypassing NAC using VLAN Hopping

Intruders exploit VLAN hopping to breach a network via Dynamic Trunking Protocol (DTP). Attackers can create a trunk with the switch by configuring the switch mode to “dynamic auto” or “dynamic desirable,” allowing DTP packets to be transmitted. The resulting trunk provides attackers with access to all VLANs.

VLANPWN

VLANPWN is an easy-to-use script created for VLAN enumeration and hopping. This tool is made up of two Python scripts aimed at making VLAN hopping easier.

1. **DoubleTagging.py:** This tool is intended to perform a VLAN-hopping attack by inserting a frame containing two 802.1Q tags and transmitting a test ICMP request.
2. **DTPHijacking.py:** This script conducts DTP-switch spoofing attacks by sending a malicious DTP-desirable frame, turning the attacker's machine into a trunk channel. This allows attackers to bypass VLAN segmentation and access all VLAN network traffic.


```
python3 DoubleTagging.py --help - Parrot Terminal
```

```
File Edit View Search Terminal Help

[~root@parrot]-(~/home/attacker/VLANPWN)
-- #python3 DoubleTagging.py --help


$ $ $ $ $SSSS$ $ $ $SSSS$ $ $ $ $ $ $ $ $
SS. SS. SS. SS. SS. SS.
$ $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$
SS $S$ $S$ SS $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$ $S$
SS $S$ $S$ SSS$. $S$ $S$ '$S,$S$ $S$ .SS::' $S$ $S$ $S$ $S$ '$S,$S$
SS $S$ $S$ SS $S$ $S$ $S$ '$S$ $S$ SS $S$ $S$ $S$ '$S$
SS ':: $S$ SS ':: $S$ ':: $S$ SS ':: :: $S$ '::
SS ':: $S$ ':: $S$ ':: $S$ SS ':: :: $S$ '::
'''''' '''''''' '''' '''' '''' '''' '''' '''' '''' ''''

VLAN Double Tagging inject tool. Jump into another VLAN!

Author: @necreasing, <necreasing@protonmail.com>

usage: DoubleTagging.py [-h] --interface INTERFACE --nativevlan NATIVEVLAN --targetvlan TARGETVLAN
                        --victim VICTIM --attacker ATTACKER

options:
  -h, --help            show this help message and exit
  --interface INTERFACE Specify your network interface
  --nativevlan NATIVEVLAN Specify the Native VLAN ID
  --targetvlan TARGETVLAN Specify the target VLAN ID for attack
  --victim VICTIM       Specify the target IP
  --attacker ATTACKER   Specify the attacker IP
```

Figure 12-18: Screenshot of VLANPWN

Bypassing NAC using Pre-Authenticated Device

Attackers can gain access to an authenticated device and use it to bypass Network Access Control (NAC). They place their device, such as a Raspberry Pi, between the pre-authenticated device and the authentication server to ensure that the traffic flows through their device.

nac_bypass_setup.sh

`nac_bypass_setup.sh` is a tool to bypass NAC using a pre-authenticated device. To bypass NAC, the fundamental necessity is to gain access to a device that has previously been authenticated. This device is utilized to connect to the network and subsequently transfer network packets from another device. This process entails positioning the attacker's system between the network switch and the authenticated device. One method to achieve this is by using a Raspberry Pi paired with two network adapters.

```
nac_bypass_setup.sh v0.6.4 usage:
-1 <eth>    network interface plugged into switch
-2 <eth>    network interface plugged into victim machine
-a          autonomous mode
-c          start connection setup only
-g <MAC>    set gateway MAC address (GWMAC) manually
-h          display this help
-i          start initial setup only
-r          reset all settings
-R          enable port redirection for Responder.py
-S          enable port redirection for OpenSSH and start the service
```

Figure 12-19: Screenshot displaying nac_bypass_setup.sh Parameters

Bypassing Endpoint Security using Ghostwriting

Ghostwriting is a technique for evading detection that entails altering the malware code's structure without impacting its performance. This is achieved by breaking down the assembly code and incorporating arbitrary code. Attackers utilize this approach to circumvent security measures on endpoint devices like antivirus software and conceal malware in order to avoid detection based on signatures.

Tools such as Ghostwriting.sh are employed by attackers to alter the structure of the malware.

Ghostwriting.sh

Ghostwriting is employed to evade antivirus programs by means of binary deconstruction, the addition of arbitrary assembly code, and subsequent reconstruction. It leverages the integrated tools within Metasploit to execute these operations. Ghostwriting.sh is a utility designed to automate this procedure.

Bypassing Endpoint Security using Application Whitelisting

Application whitelisting is a Windows security feature that allows only signed applications to run, helping to prevent malicious executions. Attackers exploit DLL hijacking by placing a malicious DLL with a legitimate name in the same directory as an executable. This enables the malicious DLL to execute with the application, potentially disabling endpoint security. By manipulating the DLL search order, attackers can maintain persistence, elevate privileges, and evade detection.

Tools like rundll32.exe, regsvr32.exe, and PowerShell can also be used to load malicious DLLs. For instance, an attacker might run a command with regsvr32.exe to execute a malicious DLL:

```
regsvr32.exe /s /n /u /i:"C:\path_to_malicious.dll"
```

Bypassing Endpoint Security by Dechaining Macros

Microsoft Office macros are frequently utilized to streamline various tasks and processes for users. Attackers can take advantage of macros to run malicious code and compromise a system. Since macros are built on VBScript, attackers develop malicious codes using VBA. In this context, the dechaining of macros is employed to bypass endpoint detection and alter memory, the registry, and

other Windows files via VBScript. This method enables attackers to avoid detection from both dynamic and static analysis techniques.

The strategies implemented to bypass endpoint security through the dechaining of macros include the following:

- Spawning through ShellCOM
- Spawning using XMLDOM
- Spawning through WmiPrvse.exe
- Creating Scheduled Tasks
- Registry Modification
- Dropping Files
- Downloading Content
- Embed File and Drop

Bypassing Endpoint Security by Clearing Memory Hooks

Memory hooking is a technique used to observe and modify how an application runs. An EDR agent implements these hooks to gather data for conducting behavior-based analyses. The hooks transmit information to the EDR agent, which operates at a high-privileged kernel level, enabling the detection of malicious activities like remote code execution, lateral movement, and privilege escalation in real time.

For example, when a new process is created in a detached state, and the memory permissions are modified to facilitate the execution of the “WriteProcessMemory” process, the EDR system can still monitor the data and assess whether the active process is harmful or not.

In such scenarios, attackers frequently attempt to evade the EDR by unhooking the EDR DLLs in memory. To achieve this, attackers must locate the application's DLLs, related functions, and exported syscalls. They utilize open-source tools such as the x64dbg debugger to pinpoint the hooked syscalls preserved in memory during execution.

Bypassing Endpoint Security by Process Injection

Process Injection is an advanced method utilized by attackers to insert malicious code into the memory of active processes. By embedding code into a currently running process, malware can avoid being detected by security software that frequently fails to monitor internal changes within each process. This technique can be a component of a broader strategy used by attackers to establish persistence, enhance privileges, or execute other harmful actions discreetly. Process injection assaults can be carried out using Windows API functions like VirtualAllocEx(), WriteProcessMemory(), and CreateRemoteThread().

- VirtualAllocEx(): Allocates memory in a target process for payload injection.
- WriteProcessMemory(): Writes data (malicious code) to the target process's memory after allocation.
- CreateRemoteThread(): Creates a new thread in the target process to execute the injected code.



Figure 12-20: Illustration of Bypassing Endpoint Security by Process Injection

Bypassing the EDR using LoLBins

Living off the Land Binaries (LoLBins) are legitimate system utilities that are either preinstalled with the operating system or can be downloaded from trusted sources like Microsoft. Malicious actors can take advantage of these LoLBins for nefarious activities, including the installation of malware or maintaining persistence within targeted networks. By utilizing LoLBins, these attackers can avoid detection by security measures since their actions seem to be normal and authorized. Additionally, these tools enable attackers to carry out various malicious tasks, such as deploying Command and Control (C2) agents for advanced control during post-exploitation without raising alerts or being identified by Endpoint Detection and Response (EDR) solutions.



Figure 12-21: Diagram showing the Attack Overview

Bypassing Endpoint Security by Control Panel (CPL) Side-Loading

CPL files are primarily designed for Windows control panels to help organize various tools and provide quick access to them, making management easier. However, attackers can exploit these CPL files to bypass detection systems by using legitimate .cpl files, which are usually linked to Windows Control Panel applets. This tactic, known as CPL sideloading, mimics the functionality of the original CPL applet, which allows malicious activities to appear harmless to both users and security systems.

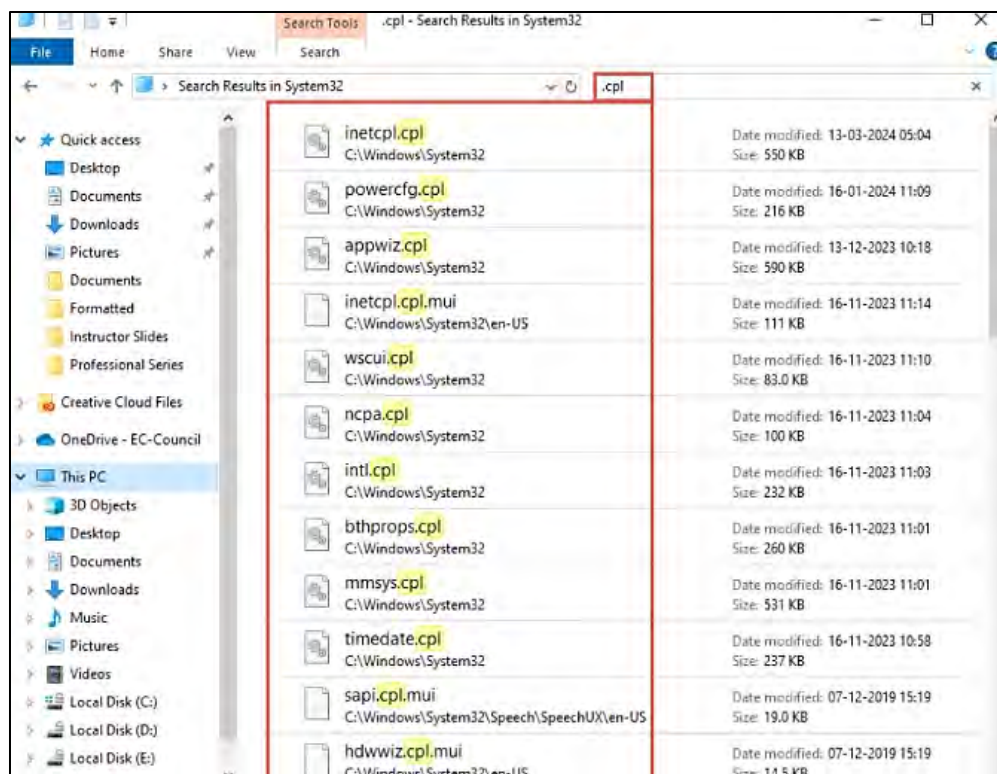


Figure 12-22: Screenshot showing the .cpl files in a Windows System

When a user executes a CPL file, malicious code can be loaded by exploiting vulnerabilities in how Windows handles these files. Attackers may trick Windows into loading the CPL file through another application, such as manipulating a trusted app's configuration. Since detection systems often overlook CPL files, attackers can maintain persistent access by ensuring the file runs at startup or at certain intervals. They may also rename harmful .dll files as .cpl and register them in the Windows registry. Even if these DLLs do not conform to CPL file standards, they can still be activated when the Control Panel is opened.



EXAM TIP: CPL sideloading serves as an evasion technique for attackers, using trusted system components to hide malicious activities and bypass detection. Attackers can use tools such as CPLResourceRunner to create malicious CPL files.

Bypassing Endpoint Security using ChatGPT

The rise of advanced technologies has resulted in the emergence of new trends where malicious individuals can leverage ChatGPT to boost the effectiveness and complexity of their operations. Given its capability to create polymorphic malware, attackers can exploit such malware to circumvent traditional endpoint security measures, making it challenging to implement mitigation strategies.

Attackers utilize ChatGPT to alter harmful code and produce various iterations for different malicious purposes. Furthermore, they can impose specific restrictions, such as modifying the employment of any API calls, which complicates detection efforts. ChatGPT is capable of converting specific code into base64, as illustrated in Figure 12-23.

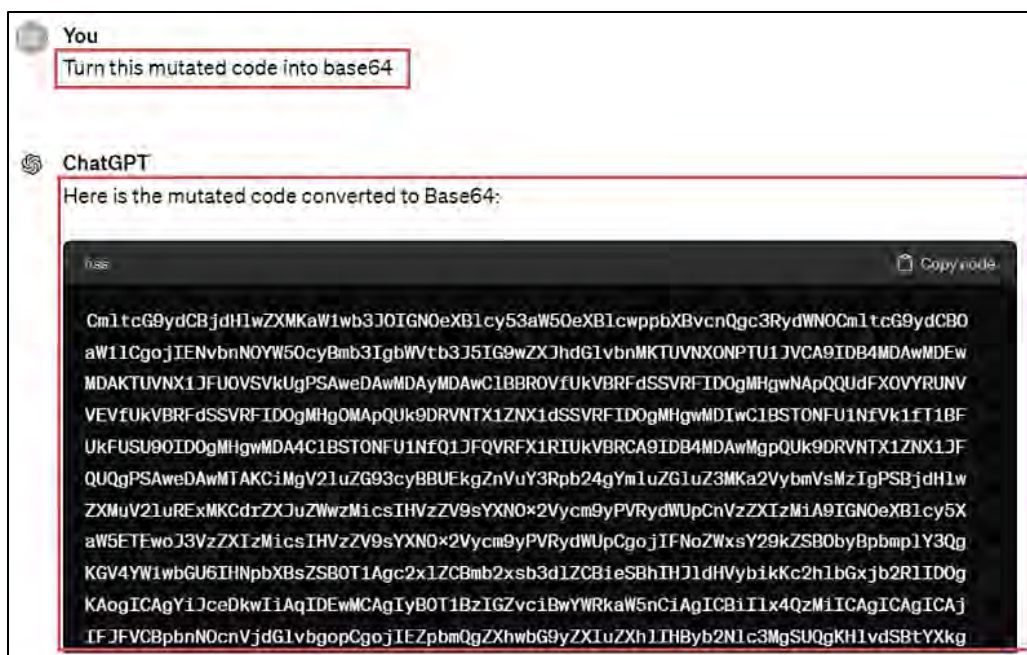


Figure 12-23: Screenshot showing based64 code of a Mutated Code

Attackers could use ChatGPT to generate injectors that can mutate when needed. By requesting code with unique parameters each time, they can create a program that effectively evades detection, making it more difficult for endpoint security systems to identify.

Bypassing Antivirus using Metasploit Templates

Attackers often try to ensure that their malicious payloads bypass antivirus software on the victim's machine. Metasploit templates can be used in this regard.

Bypassing Windows Antimalware Scan Interface (AMSI)

Antimalware Scan Interface (AMSI) is an API in Windows that improves malware protection for applications. It can work alongside compatible antimalware solutions on the system to boost detection abilities, which include both signature and reputation-based identification. Attackers can bypass Windows AMSI by altering elements like URLs, functions, or internal files. The methods employed to evade AMSI are as follows:

- PowerShell Downgrade
- Obfuscation
- Forcing an error
- Memory Hijacking

Other Techniques for Bypassing Endpoint Security

Attackers employ various evasion techniques to maintain persistence on compromised systems while avoiding detection by sandboxing services, User Behavior Analytics (UBA), or Security Information and Event Management (SIEM) solutions that generate behavior-based alerts. After compromising a system, they evade different security controls in place to maintain stealth and expand their malicious activities.

Although organizations may utilize various security measures such as Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), or Endpoint Detection and Response (EDR) tools, attackers often implement sophisticated techniques to conceal their activities and remain undetected. Consequently, to bypass behavior-based EDR tools, attackers exploit advanced mechanisms and sophisticated malware to mask their malicious operations.

Hosting Phishing Sites on Popular Infrastructure

The EDR mechanism in organizations blocks IP addresses involved in phishing and malicious activities using regularly updated blacklists. However, attackers exploit this by hosting phishing websites on reputable cloud services like Google Cloud and AWS, which are not blacklisted. This allows them to use these platforms as command and control servers. Additionally, attackers can distribute malware through popular social media by embedding malicious code in images using steganography, enabling infected systems to evade endpoint security by reading hidden instructions.

Passing Encoded Commands

Attackers can use encrypted commands to evade detection mechanisms under certain conditions. For example, by sending Base64 encoded commands, they can obscure their arguments and code. Additionally, attackers may utilize hex-format encryption to ping various IP addresses, helping them avoid detection by security systems.

Fast Flux DNS Method

The fast flux technique enables attackers to swiftly change both DNS names and IP addresses and is commonly employed by extensive botnets. This method assists attackers in evading numerous security controls. Additionally, it enables the attacker to bypass blacklists and conceal the Command and Control (C&C) server behind compromised machines acting as reverse proxies. In this scenario, a victim's system will only connect to the fast flux agents rather than the actual C&C server.

Timing-based Evasion

This technique involves executing malware based on specific timing or following particular actions taken by the victim. These actions might include opening a certain window and clicking on it, which triggers the execution after the system has restarted. Additional examples of this include sleep patching, delay APIs, and time bombs.

Signed Binary Proxy Execution

This method enables attackers to utilize trusted system utilities to run malicious code, thereby evading EDR solutions. Attackers exploit these legitimate utilities because they are backed by digital certificates, facilitating the proxy execution of harmful code. For instance, they might leverage rundll32 to execute harmful commands.

Shellcode Encryption

Attackers can transmit encrypted commands that carry malicious payloads to avoid detection by EDR solutions. For example, they may encrypt shellcode using encryption techniques like XOR or RC4, often embedding decryption keys within the malware or generating them on the fly. When an encrypted command is executed, the malware decrypts the shellcode and runs it in memory. Additionally, attackers might use AES encryption to disguise the static signatures of the shellcode.

Reducing Entropy

In this method, attackers alter the characteristics of a binary to make it seem less suspicious to security solutions. For instance, they can decrease entropy by adding low-entropy resources, such as low-entropy images, into the binary. Attackers may also diminish entropy by integrating strings, like `chrome.dll`, from system files.

Escaping the (Local) AV Sandbox

EDR solutions typically assess a binary's behavior within a local sandbox environment for a limited duration, usually just a few seconds, to reduce the impact on the user experience. Attackers take advantage of this limitation by postponing shellcode execution. For example, they can compute large prime numbers and use them as keys to prolong the shellcode encryption process, enabling them to evade sandbox analysis constraints.

Disabling Event Tracing for Windows (ETW)

Many EDR solutions, particularly Microsoft Defender for Endpoints, depend on Event Tracing for Windows (ETW) for process and system call tracking. The ETW includes kernel and userland components, specifically within `ntdll.dll`. Attackers can compromise ETW by patching the `EtwEventWrite` function to return a success status.

Direct System Calls and Evading “Mark of the Syscall”

Attackers can evade `ntdll.dll` hooks by using direct system calls, such as calling `NtAllocateVirtualMemory` instead of `VirtualAlloc`. They can also retrieve syscall IDs and invoke system calls directly.

Spoofing the Thread Call Stack

To avoid detection while dormant, implants can obscure their thread return addresses, typically pointing to the shellcode. By intercepting the `Sleep()` function, attackers overwrite the return address and later restore it after `Sleep()` returns.

In-memory Encryption of Beacon

Attackers can also encrypt the implant's executable memory during dormancy by using a beacon hook like `Sleep()`. They locate and encrypt the shellcode while sleeping, then decrypt it upon waking, allowing them to evade detection by security solutions.

IDS/Firewall Evading Tools

During firewall evasion, attackers utilize various security auditing tools to evaluate how firewalls operate. This section highlights some of these tools that assist attackers in bypassing firewall restrictions. These tools automate the process of bypassing firewall rules, improving effectiveness and reducing the time required.

Traffic IQ Professional

Traffic IQ Professional is a tool designed to evaluate and verify the functionality of security devices by creating standard applications or attack traffic between two virtual machines. Typically utilized by security professionals, this tool is for assessing, auditing, and testing the behavioral traits of non-proxy packet-filtering devices, which may include application firewalls, IDS, IPS, routers, switches, and more. Nonetheless, since this tool can produce customized attack traffic, it is also frequently used by attackers to evade the security devices installed at the network's perimeter.

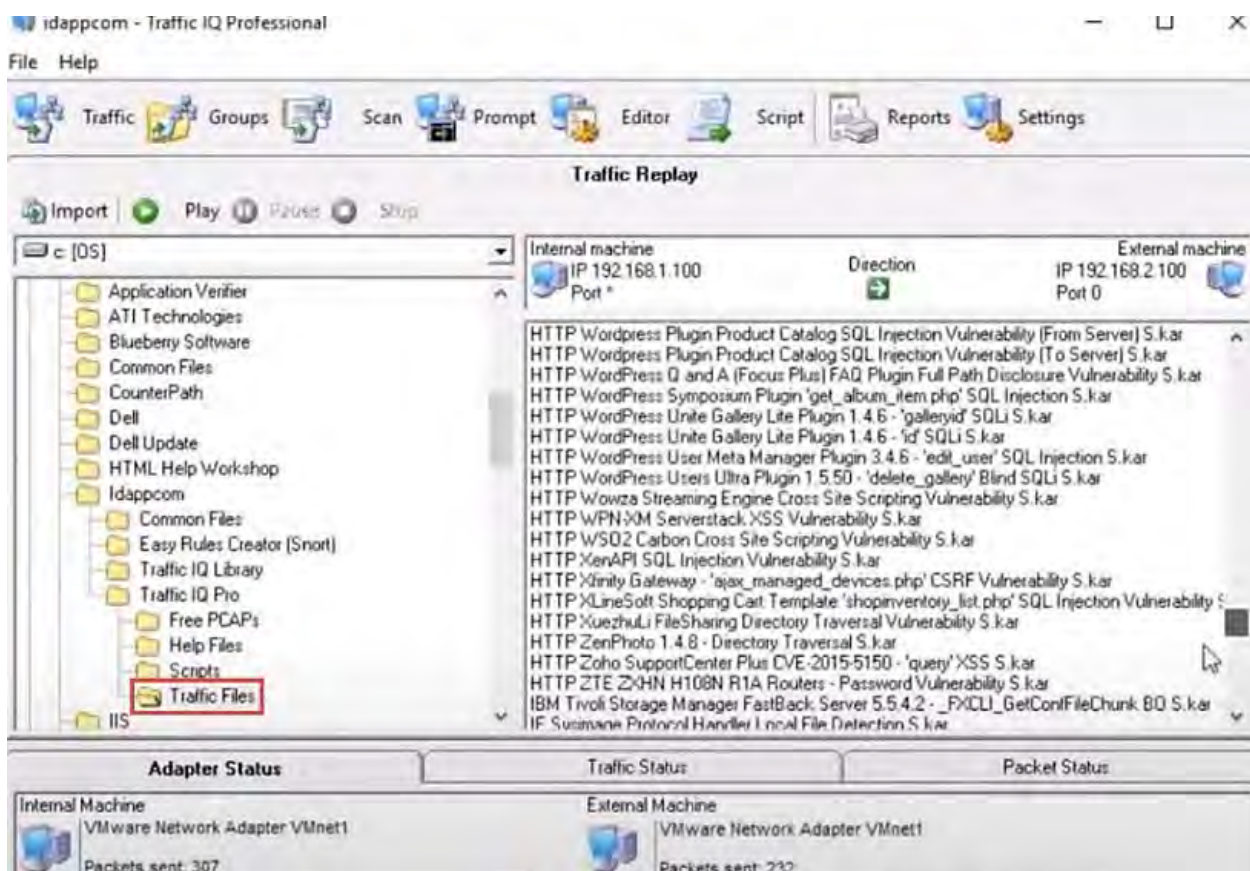


Figure 12-24: Traffic IQ Professional

Some additional IDS/firewall evasion tools are as follows:

- Nmap
- Metasploit
- PingRAT
- KoviD

- Green Tunnel
- Hyperion

Packet Fragment Generator Tools

There are various packet fragment generators that attackers use to perform fragmentation attacks on firewalls to bypass them.

Colasoft Packet Builder

Colasoft Packet Builder allows users to generate tailored network packets and split packets into fragments. Malicious actors utilize this software to craft specific harmful packets and fragment them to avoid detection by firewalls. It enables the creation of custom network packets, including Ethernet, ARP, IP, TCP, and UDP packets. On the other hand, security experts make use of this tool to assess the defensive capabilities of networks against potential attacks and intrusions.

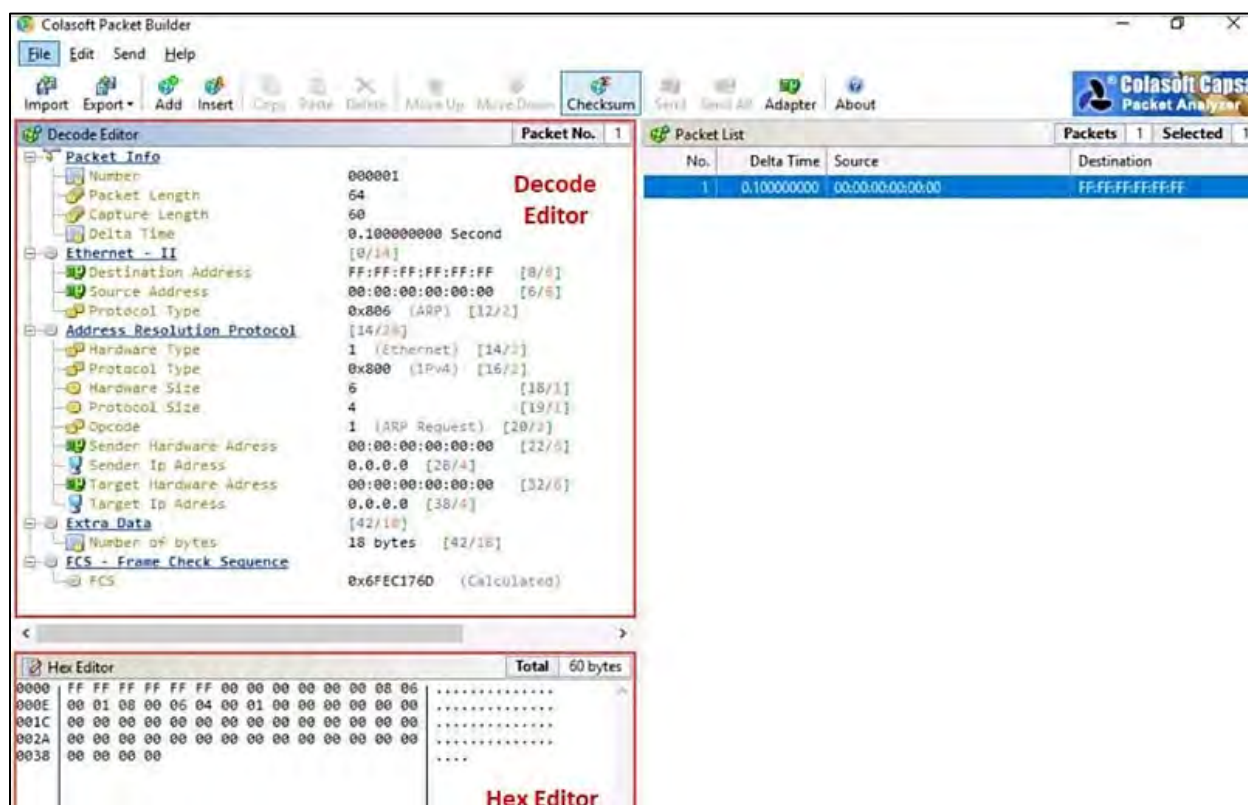


Figure 12-25: Screenshot of Colasoft Packet Builder

Some additional packet generator tools are listed below:

- NetScanTools Pro
- CommView
- Ostinato
- WAN Killer
- WireEdit

Honeypot Concepts

Honeypots are crucial elements of cybersecurity strategies. They function as decoy systems designed to attract and detect malicious activities. Attackers may use various tools and techniques to identify honeypots while trying to infiltrate a target network. This section will explore different concepts and techniques related to honeypots, particularly focusing on how they can be detected within a target network.

Honeypot

Honeypots are devices or systems deployed to trap attackers attempting to obtain unauthorized access to a system or network. They are deployed in an isolated environment and are monitored. Typically, honeypots are deployed in DMZ and configured identically to a server. Any probe, malware, or infection will be immediately detected as the honeypot appears to be a legitimate part of the network.

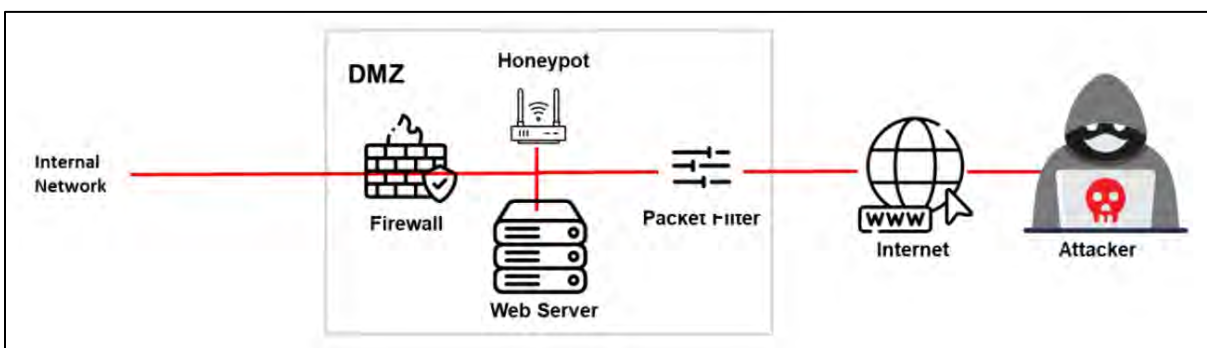


Figure 12-26: Example of Honeypot

Types of Honeypots

High-Interaction Honeypots

High-interaction honeypots are configured with various services that are enabled to waste an attacker's time to obtain information about the intrusion. Multiple honeypots can be deployed on a single physical machine and can be restored if an attacker even compromises the honeypot.

Medium-interaction Honeypots

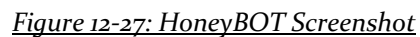
Medium-interaction honeypots emulate a real OS along with target network applications and services, allowing for more complex attack logging and analysis than low-interaction honeypots. They capture more useful data but respond only to preconfigured commands, increasing the risk of intrusion. However, a key drawback is that attackers can quickly identify abnormal system behavior.

Low-Interaction Honeypots

Low-interaction honeypots are configured to entertain only the services that users commonly request. Response time, less complexity, and the need for few resources make low-interaction honeypot deployment easier compared to high-interaction honeypots.

Honeypots are security tools that enable the security community to observe the tactics and exploits used by attackers by recording all their actions, allowing for a swift response to such exploits before the attacker can take advantage of or compromise the system.

HoneyBOT is a medium-interaction honeypot designed for Windows. A honeypot establishes a secure setting to collect and engage with uninvited traffic on a network. HoneyBOT is a user-friendly solution that is perfect for research in network security or as a component of a proactive intrusion detection system.



- Blumira honeypot software
- NeroSwarm Honeypot
- Valhala Honeypot
- Cowrie
- StingBox

Detecting Honeypots

The basic logic of Detecting a Honeypot in a network is probing the services. An attacker usually crafts a malicious packet to scan the services running on a system and opens and closes the port information. These services may be HTTPS, SMTPS, IMAPS, or something else. Once an attacker extracts the information, he/she can attempt to build a connection; the actual server will complete the three-way handshake process, but denying a handshake indicates the presence of a honeypot. Send-Safe Honeypot Hunter, Nessus, and Hping tools can be used to detect honeypots.

Detecting and Defeating Honeypots

A honeypot is a security tool designed to counterattack and ensnare intruders. Honeypots attract attackers into carrying out malicious actions, and the data from these attacks offers valuable insights into the types and severity of threats that a network may encounter. For an attacker, it is crucial to ascertain whether the target system is genuine or a honeypot to successfully breach the network without being discovered. Recognizing and overcoming these honeypot setups discreetly is a primary objective for a skilled hacker.

Below are some methods used to identify, detect, and neutralize different honeypot structures:

- Detecting the presence of Layer 7 Tar Pits
- Detecting the presence of Layer 4 Tar Pits
- Detecting the presence of Layer 2 Tar Pits
- Detecting Honeypots running on VMware
- Detecting the presence of Honeyd Honeypot
- Detecting the presence of User-Mode Linux (UML) Honeypot
- Detecting the presence of Snort_inline Honeypot
- Detecting the presence of Fake AP
- Detecting the presence of Bait and Switch Honeypots

Honeypot Detection Tools

Attackers utilize honeypot detection tools such as Send-Safe Honeypot and SniffingBear to detect honeypots in the target organizational networks.

Send-Safe Honeypot Hunter

Send-Safe Honeypot Hunter is a tool used to verify lists of HTTPS and SOCKS proxies to identify potential honeypots.

The tool can:

- Check lists of HTTPS, SOCKS4, and SOCKS5 proxies with any ports
- Check several remote or local proxylists at once
- Upload "Valid proxies" and "All except honeypots" files to FTP
- Process proxylists automatically in every specified period
- Be used for usual proxylist validating as well

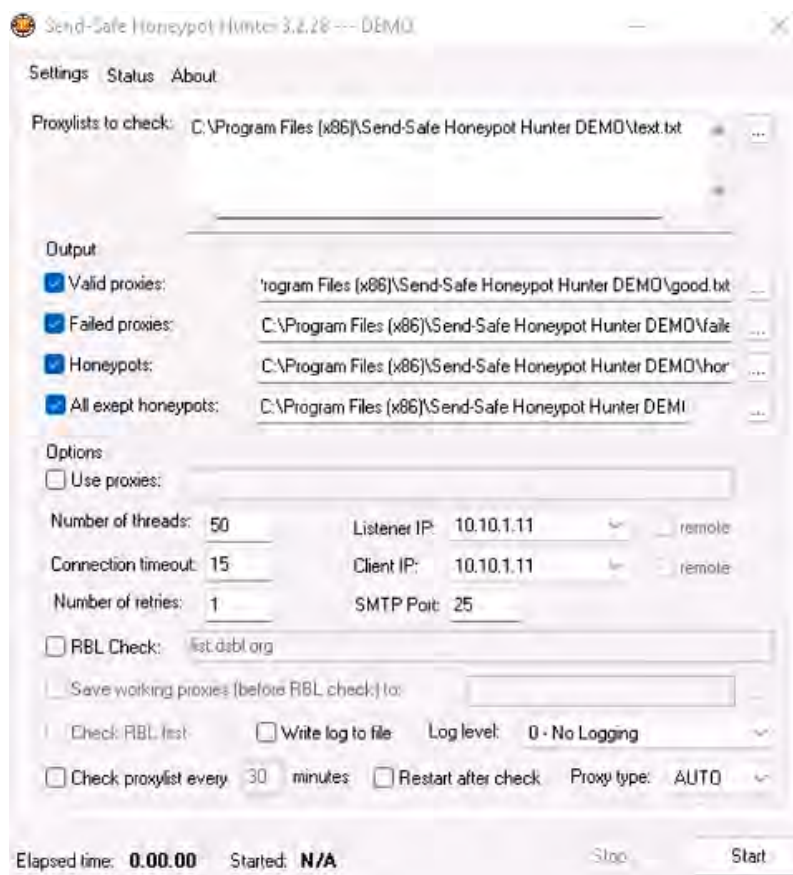


Figure 12-28: Screenshot of Send-Safe HoneyPot Hunter

IDS/Firewall Evasion Countermeasures

Managing and preventing an evasion technique is a great challenge. However, many techniques make it difficult for an attacker to evade detection. These defensive and monitoring techniques ensure the detection system protects the network and provides more control of traffic. Some of these techniques are basic troubleshooting and monitoring, whereas other techniques focus on the proper configuration of IPS/IDS and firewalls. Initially, observe and troubleshoot the firewall by:

- Port scanning
- Banner grabbing
- Firewalking
- IP address spoofing
- Source routing
- Bypassing firewall using IP in URL
- Attempting a fragmentation attack
- Troubleshooting behavior using proxy servers
- Troubleshooting behavior using ICMP tunneling

Shutting down the unused ports associated with known attacks is an effective step in preventing evasion. Other effective steps include performing in-depth analysis, resetting the malicious session, updating patches, deploying IDS, normalizing fragmented packets, increasing TTL expiry, blocking TTL expired packets, reassembling packets at the IDS, strengthening security, and correctly enforcing policies.

How to Defend Against IDS Evasion

- Shut down ports linked to known attack hosts.
- Analyze ambiguous network traffic for threats.
- Use TCP FIN or RST packets to terminate malicious sessions.
- Look for nop opcodes other than 0x90 to combat polymorphic shellcode.
- Train users to recognize attack patterns and maintain system updates.
- Deploy IDS after analyzing network topology and traffic.
- Use a traffic normalizer to clarify packet streams for the IDS.
- Ensure IDS can normalize and reassemble fragmented packets.
- Define DNS servers in routers or similar devices.
- Strengthen the security of modems and routers.
- Block ICMP TTL expired packets at the external interface and adjust TTL values.
- Regularly update antivirus signatures.
- Use traffic normalization in the IDS to prevent evasion.
- Store attack data for future analysis.
- Ensure packets come from IDS-secured paths; analyze non-IDS packets.
- Properly configure snort rules to prevent DoS attacks and false positives.
- Check for malicious script injections in the snort rules directory.
- Use hybrid signature-based techniques with behavioral analysis for zero-day exploits.
- Configure IDS rules to detect tunneling and obfuscation methods.
- Conduct behavioral analysis to detect polymorphic behavior.
- Drop packets with invalid or spoofed IPs.
- Deploy high-interaction honeypots with realistic services.
- Configure IDS to handle overlapping fragments and reassemble fragmented sessions.

How to Defend Against Firewall Evasion

- Configure the firewall to filter out intruder IP addresses and deny all traffic except essential services.
- Use a unique user ID for running firewall services instead of the administrator/root ID.
- Set up a remote syslog server and implement strict security measures.
- Regularly monitor firewall logs and investigate suspicious entries.
- Disable all FTP connections by default.
- Catalog and review inbound/outbound traffic through the firewall.
- Conduct regular risk assessments of firewall rules.
- Control user access to firewall configurations.
- Specify source/destination IP addresses and ports.

- Notify the security policy administrator of firewall changes and document them.
- Control physical access to the firewall and take regular backups.
- Schedule regular security audits and implement HTTPS/TLS inspections.
- Use tools like HTTP Evader for automated testing of firewall evasions.
- Employ deep packet inspection and set connection limits per IP address.
- Restrict traffic based on geographical regions and user behavior.
- Adopt a zero-trust security model for all network traffic.

How to Defend Against Endpoint Security Evasion

- Use advanced antivirus solutions with behavioral analysis, machine learning, and cloud-based threat intelligence.
- Keep all operating systems and applications updated.
- Implement network segmentation to contain malware spread.
- Implement the least privilege principle for user access.
- Enable Multifactor Authentication (MFA) for critical system access.
- Use secure remote access methods, like VPNs.
- Deploy honeypots to deceive attackers.
- Maintain detailed logs of network activities and analyze them with SIEM solutions.
- Utilize application whitelisting to restrict program execution.
- Employ code signing certificates for software authenticity.
- Conduct regular security audits and penetration tests.
- Use MAC address filtering for device authorization.
- Monitor the network in real time for unusual activities.
- Implement security techniques like ASLR, DEP, and CFI for memory protection.
- Use File Integrity Monitoring (FIM) for critical files.
- Enforce USB security measures to prevent malware introduction.
- Deploy DLP tools to oversee sensitive data movement.
- Restrict sensitive information transfer via removable media or email.

How to Defend Against NAC Evasion

- Create thorough policies that explicitly outline access control measures, authentication criteria, and action steps for non-compliant devices.
- Implement device profiling to recognize and categorize devices attempting to connect to the network, which aids in identifying unauthorized devices or those displaying unusual activity.
- Ensure that NAC policies are applied dynamically, enabling real-time modifications based on evolving security needs or identified threats.
- Mandate Multifactor Authentication (MFA) for accessing sensitive network resources to reduce unauthorized access risk, even if a password is compromised.
- Employ Role-Based Access Control (RBAC) to restrict resource access based on user roles, thereby decreasing the likelihood of lateral movement by attackers.

- Isolate sensitive network resources into distinct segments to minimize the effects of NAC evasion incidents.
- Utilize Virtual Local Area Networks (VLANs) to confine devices to designated network segments.
- Perform frequent vulnerability assessments and security audits to uncover potential vulnerabilities in the NAC system.
- Implement MAC address filtering to permit only authorized devices onto the network.
- Embrace a zero-trust model whereby every device and user must be authenticated at every access point.
- Deploy endpoint security solutions that encompass antimalware, antiphishing, and firewall defenses.
- Use RBAC to uphold the principles of least privilege access.
- Exchange threat intelligence with the community to bolster collective defense against NAC evasion and other cyber threats.
- Integrate NAC solutions with SIEM platforms to connect NAC events with other security telemetry data, such as logs from firewalls, IDS/IPS, and endpoint security solutions.
- Implement advanced authentication methods such as certificate-based or biometric authentication to strengthen NAC security.
- Actively pursue and investigate NAC evasion attempts through efficient threat-hunting initiatives.

How to Defend Against Anti-virus Evasion

- Use behavior-based antivirus solutions with machine learning to detect unknown threats.
- Regularly update antivirus software for the latest threat intelligence and real-time protection.
- Implement Endpoint Protection (EPP) with EDR capabilities for enhanced security.
- Segment networks to limit access to sensitive systems and prevent malware spread.
- Raise awareness about the risks of downloading from untrusted sources and opening unknown attachments.
- Keep applications and operating systems updated to patch vulnerabilities.
- Use application whitelisting to block unauthorized programs and reduce attack surfaces.
- Monitor for suspicious activities like file modifications and process injections.
- Execute potential threats in sandbox environments to observe behavior safely.
- Deploy SIEM solutions for centralized log collection and abnormal activity detection.
- Analyze scripts in real-time to prevent malicious documents.
- Combine security layers like firewalls, IDS, and IPS for stronger protection.
- Utilize signature-less detection methods like anomaly detection to identify system irregularities.
- Limit user privileges to essential levels to mitigate malware risks.
- Utilize secure remote access methods, such as VPNs, to protect against external attacks.

Summary

This chapter covered various concepts and solutions related to Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS), and firewalls. It explained different techniques for bypassing IDS and firewalls, as well as methods to evade Network Access Control (NAC) and endpoint security. Additionally, this chapter introduced various tools used for IDS and firewall evasion. It also included an overview of honeypot concepts and strategies for detecting and mitigating honeypots. Finally, the chapter concluded with a detailed discussion of various countermeasures that can be implemented to prevent evasion attempts by threat actors targeting IDS and firewalls.

Mind Map



Figure 12-29: Mind map of Evading IDS, Firewalls, and Honeypots

Practice Questions

1. Which of the following is a primary difference between IDS and IPS?
 - A. IDS actively blocks threats, while IPS only generates alerts
 - B. IDS only detects threats, while IPS can detect and prevent threats
 - C. IDS is a type of firewall, while IPS is an endpoint security tool
 - D. IDS uses signatures, whereas IPS uses anomaly detection only
2. Which of the following tools can be used for IDS/Firewall evasion?
 - A. Snort
 - B. Metasploit
 - C. Nmap
 - D. Both B and C
3. Which type of honeypot is designed to closely mimic a real production system?
 - A. Low-interaction honeypot
 - B. Medium-interaction honeypot
 - C. High-interaction honeypot
 - D. Hybrid honeypot
4. Which of the following is a countermeasure against IDS evasion techniques?
 - A. Enabling Deep Packet Inspection (DPI)
 - B. Disabling logging on critical devices
 - C. Using a single firewall rule for all traffic
 - D. Reducing network monitoring capabilities
5. The _____ technique involves modifying packet structure to bypass IDS detection.
 - A. Obfuscation
 - B. Fragmentation
 - C. Tunneling
 - D. Redirection
6. Honeypots help security teams by acting as a _____ to attract and analyze attacks.
 - A. Firewall
 - B. Decoy
 - C. Proxy
 - D. Sandbox
7. Stateful firewalls inspect only the header of packets to make filtering decisions.
 - A. True
 - B. False
8. IDS evasion techniques include fragmentation, protocol obfuscation, and polymorphic payloads.
 - A. True
 - B. False
9. What is the primary function of a Web Application Firewall (WAF)?

- A. Detect and prevent DDoS attacks
 - B. Filter and monitor HTTP/S traffic for malicious activity
 - C. Encrypt all web communications
 - D. Provide endpoint security protection
10. What is a common characteristic of low-interaction honeypots?
- A. They fully emulate a real operating system
 - B. They require extensive maintenance and security updates
 - C. They provide limited interaction to capture basic attack patterns
 - D. They provide attackers with full administrative access
11. Which of the following is an effective countermeasure against IDS evasion techniques like fragmentation?
- A. Disabling Deep Packet Inspection
 - B. Normalizing fragmented packets before analysis
 - C. Limiting log storage to conserve disk space
 - D. Filtering traffic only at the network perimeter
12. Which of the following is a key benefit of a Network-based Intrusion Detection System (NIDS) over a Host-based IDS (HIDS)?
- A. Can analyze encrypted traffic
 - B. Monitors application logs directly
 - C. Provides better protection against insider threats
 - D. Detects attacks before they reach the target system
13. Which type of IDS alert occurs when the system correctly detects a real attack?
- A. True Positive
 - B. False Positive
 - C. True Negative
 - D. False Negative
14. How does HTML smuggling bypass network security defenses like firewalls and web proxies?
- A. By using encrypted network packets
 - B. By embedding malicious code in an HTML blob that is processed client-side
 - C. By modifying firewall rules through privilege escalation
 - D. By launching direct brute-force attacks on the network
15. Which type of IDS is primarily targeted by an insertion attack?
- A. Behavior-based IDS
 - B. Signature-based IDS
 - C. Host-based IDS (HIDS)
 - D. Machine learning-based IDS
16. Using VPNs or encrypted tunnels can help attackers bypass IDS monitoring.
- A. True
 - B. False

17. Which of the following components is manipulated in an HTML smuggling attack to deliver malware?
- A. MIME type and JavaScript blob
 - B. DNS records
 - C. HTTP headers only
 - D. TCP handshake process
18. A False Positive in IDS means that the system has:
- A. Correctly detected a real attack
 - B. Missed detecting a real attack
 - C. Incorrectly flagged legitimate activity as an attack
 - D. Completely ignored network traffic
19. Which protocol is commonly exploited in VLAN hopping attacks?
- A. Dynamic Trunking Protocol (DTP)
 - B. Simple Network Management Protocol (SNMP)
 - C. Border Gateway Protocol (BGP)
 - D. Open Shortest Path First (OSPF)
20. Which switch configuration makes a VLAN vulnerable to VLAN hopping attacks?
- A. "Dynamic Auto" or "Dynamic Desirable" mode enabled
 - B. "Access Mode" manually configured on all ports
 - C. "Shutdown" mode enabled on all unused ports
 - D. Static VLAN assignment with trunking disabled
21. What network behavior might indicate the presence of a honeypot?
- A. Denying or delaying a three-way handshake
 - B. Responding to ARP requests too quickly
 - C. Accepting all incoming connections without authentication
 - D. Routing traffic to multiple VLANs
22. Firewalking is a technique used to determine firewall rule sets and identify open ports.
- A. True
 - B. False
23. What happens when an IDS mistakenly rejects a malicious packet?
- A. The IDS fails to detect the attack
 - B. The IDS blocks all further traffic from the attacker
 - C. The IDS automatically updates its signatures
 - D. The IDS alerts the network administrator immediately
24. Why do fragmented packets help in IDS evasion?
- A. The IDS may not properly reassemble the fragments for inspection
 - B. The IDS automatically blocks fragmented packets
 - C. The end system rejects fragmented packets
 - D. Fragmented packets increase network speed
25. Which type of IDS alert is the most dangerous because an actual attack goes undetected?

- A. True Positive
- B. False Positive
- C. True Negative
- D. False Negative

Answers

1. Answer: B

Explanation: An Intrusion Detection System (IDS) monitors network traffic for suspicious activity and generates alerts but does not take action to block threats. In contrast, an Intrusion Prevention System (IPS) not only detects threats but can actively prevent them by blocking or mitigating malicious traffic in real time.

2. Answer: D

Explanation: Both Metasploit and Nmap can be used for IDS/Firewall evasion techniques. Metasploit has various features, such as obfuscating payloads and utilizing different techniques to bypass firewalls and IDS systems. Nmap can use techniques like fragmentation, decoy scans, and other evasion methods to bypass detection by firewalls or IDS systems.

3. Answer: C

Explanation: A High-interaction honeypot is designed to closely mimic a real production system. It creates a more convincing and complex environment that can engage attackers for longer periods, allowing for detailed analysis of their tactics. This type of honeypot provides more opportunities to observe attacker behavior, though it requires more resources to maintain and secure.

4. Answer: A

Explanation: Deep Packet Inspection (DPI) is a countermeasure against IDS evasion techniques because it inspects the actual contents of network packets beyond just their headers. DPI can detect malicious payloads or hidden data that might bypass simpler methods like basic packet filtering or shallow inspection.

5. Answer: B

Explanation: The Fragmentation technique involves breaking packets into smaller fragments, which may bypass IDS systems that do not properly reassemble the fragments or inspect them thoroughly. By fragmenting malicious packets, attackers can potentially evade detection by the IDS.

6. Answer: B

Explanation: Honeypots act as a decoy to attract attackers, allowing security teams to observe and analyze attack methods and behaviors in a controlled environment. This helps in understanding threats and improving defense strategies. Honeypots are intentionally vulnerable systems designed to lure attackers away from real, critical assets.

7. Answer: B

Explanation: Stateful firewalls track the state of active connections and make filtering decisions based on the context of the traffic (such as the state of the connection), not just the header information. They examine both the header and the packet's state in the context of the connection to ensure that packets are part of a valid session.

8. Answer: A

Explanation: IDS evasion techniques indeed include fragmentation, protocol obfuscation, and polymorphic payloads. These techniques are used to evade detection by IDS systems by altering the structure or appearance of malicious traffic.

9. Answer: B

Explanation: A WAF specifically focuses on protecting web applications by inspecting and filtering HTTP/HTTPS traffic. It is designed to identify and block common web application attacks like SQL injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).

10. Answer: C

Explanation: Low-interaction honeypots provide a minimal and controlled environment for attackers, offering limited interaction to capture basic attack patterns without fully emulating a real system. These honeypots are easier to deploy and maintain than high-interaction honeypots but are less effective for studying advanced attack behaviors.

11. Answer: B

Explanation: To effectively counteract IDS evasion techniques like fragmentation, normalizing fragmented packets before analysis ensures that the IDS can properly reassemble fragmented packets and inspect them as a whole. This helps prevent attackers from bypassing detection by sending fragmented malicious packets.

12. Answer: D

Explanation: NIDS is positioned on the network to monitor traffic across various devices and segments. It can detect attacks before they reach the target system by analyzing network traffic in real time. This allows NIDS to identify threats early in the attack lifecycle.

13. Answer: A

Explanation: A True Positive occurs when the IDS correctly detects a real attack, meaning the alert generated accurately represents a legitimate threat.

14. Answer: B

Explanation: HTML smuggling bypasses network security defenses like firewalls and web proxies by embedding malicious code within an HTML file (often in non-visible parts of the document) that is processed client-side. This makes it difficult for network security devices to detect the threat because the malicious payload is not directly visible in network traffic.

15. Answer: B

Explanation: An insertion attack targets a Signature-based IDS by introducing malicious traffic that does not match any known signature, effectively evading detection. In an insertion attack, the attacker sends malformed or obfuscated packets designed to bypass the IDS's signature matching mechanism, often by inserting data that disrupts the signature comparison process.

16. **Answer:** A

Explanation: Using VPNs or encrypted tunnels can help attackers bypass IDS monitoring because the encrypted traffic makes it difficult for the IDS to inspect the contents of the data. Since the traffic is encrypted, an IDS may only be able to see the metadata (e.g., source/destination IPs) and not the actual payload, allowing the attacker to potentially avoid detection.

17. **Answer:** A

Explanation: In an HTML smuggling attack, the attacker manipulates the MIME type and JavaScript blob to deliver malware. By crafting an HTML file with an unexpected MIME type or using JavaScript to load the malicious content, the attacker can bypass security mechanisms like firewalls and web proxies. The content is processed client-side, making it difficult for network defenses to detect the malicious payload.

18. **Answer:** C

Explanation: A False Positive in an IDS occurs when the system incorrectly identifies legitimate, non-malicious activity as an attack. This leads to unnecessary alerts and can result in wasted resources and time while investigating or mitigating non-existent threats.

19. **Answer:** A

Explanation: The Dynamic Trunking Protocol (DTP) is commonly exploited in VLAN hopping attacks. DTP is used to negotiate trunk links between switches. Attackers can manipulate DTP to establish a trunk connection where VLANs can be carried across unauthorized switches, allowing them to bypass VLAN segmentation and access traffic from different VLANs.

20. **Answer:** A

Explanation: "Dynamic Auto" and "Dynamic Desirable" modes on a switch port make it vulnerable to VLAN hopping attacks because they allow a switch to automatically negotiate a trunk link. An attacker can exploit this by sending crafted frames to force a trunk link to be established, even on ports that should be limited to a single VLAN.

21. **Answer:** C

Explanation: A honeypot is a decoy system designed to attract and observe attackers. A common behavior that might indicate the presence of a honeypot is accepting all incoming connections without authentication. This is because honeypots are often configured to appear open and vulnerable in order to lure attackers, with little or no security controls.

22. **Answer:** A

Explanation: Firewalking is a technique used to determine the rule sets of a firewall and identify open ports by sending packets with a TTL value set to expire at the firewall. This technique can reveal how a firewall is configured by analyzing how it handles specific types of traffic and which ports are open or closed. It essentially "walks" through the firewall's filtering rules and helps attackers map the network.

23. **Answer:** A

Explanation: When an IDS mistakenly rejects a malicious packet, it means that the IDS has failed to correctly identify the attack, so the malicious packet is not flagged or logged as a threat. As a result, the IDS fails to detect the attack, allowing the malicious activity to proceed without being noticed.

24. **Answer:** A

Explanation: Fragmented packets can help evade detection by an IDS because the IDS may not properly reassemble the fragments for inspection. If an attacker fragments a malicious packet into smaller parts, the IDS might only inspect the fragments individually without reassembling them, missing the malicious payload that is only apparent when the fragments are combined.

25. **Answer:** D

Explanation: A False Negative occurs when the IDS fails to detect an actual attack, meaning that a real threat goes undetected. This is the most dangerous type of alert because the system is unaware of the attack, allowing it to continue without intervention.