Advanced Networking Exercises

Legend:

Input Command

Output of the previous command

Prerequisites

- Windows Server
- Linux Ubuntu 22.04 Server

Expectations

- Complete the Tasks
- Take Screenshots of every input and output
- Explain in your own words what is happening

What you need to know

Firewalls

Firewalls serve as the first line of defense in network security, functioning as a gatekeeper that monitors and controls the flow of data between your internal network and external sources. This essential security mechanism is designed to prevent unauthorized access while allowing legitimate traffic to pass through. By scrutinizing every packet of data that attempts to enter or exit the network, firewalls play a crucial role in defending against a wide range of cyber threats, including viruses, worms, ransomware, and hacker attacks.

The Role of Firewalls in Network Security

The core function of a firewall is to implement a set of security rules that define what types of traffic are permitted or blocked. These rules are based on various criteria, such as IP addresses, domain names, protocols, ports, and even specific keywords within the data. By applying these rules, firewalls can effectively distinguish between safe and potentially harmful traffic, ensuring that only trusted data is allowed to enter the network while malicious or unnecessary traffic is kept out.

Firewalls are particularly important for businesses and organizations of all sizes. They not only protect sensitive data from external threats but also enforce network policies and prevent unauthorized access to internal resources. For instance, a company might configure its firewall to

block access to social media sites during working hours or to restrict access to sensitive areas of the network to certain employees.

Types of Firewalls

Firewalls can be broadly classified into two categories: hardware and software, with a subset being appliance firewalls. Each type serves a specific purpose and offers distinct advantages and challenges.

Hardware Firewalls are standalone physical devices placed between the network and the gateway to the internet. They are typically deployed at the perimeter of the network and are capable of handling large volumes of traffic. Hardware firewalls are particularly beneficial for organizations as they provide a robust level of protection with minimal impact on the performance of individual computers within the network.

Software Firewalls are installed and run on individual computers or servers. These firewalls offer more granular control over the traffic to and from the specific device on which they are installed. Software firewalls are ideal for providing personalized protection for individual devices, allowing users to tailor the security settings to meet their specific needs.

Appliance Firewalls combine the characteristics of hardware and software firewalls, offering a dedicated device that comes pre-installed with firewall software. These appliances are designed for ease of use and deployment, providing a balanced solution for organizations looking for both performance and flexibility.

Key Firewall Technologies

To enhance their effectiveness, modern firewalls incorporate advanced technologies and features, such as:

- **Stateful Inspection**: Unlike simpler methods that examine individual packets in isolation, stateful inspection tracks the state of active connections and makes decisions based on the context of the traffic. This allows the firewall to understand the history of a connection and apply more sophisticated filtering rules.
- **Deep Packet Inspection (DPI)**: This technology enables the firewall to examine the data within the packets themselves, not just the headers. DPI allows for detailed analysis of the content of communications, enabling the identification and blocking of specific types of content, such as malware or unauthorized data exfiltration.
- **Application Awareness**: Next-Generation Firewalls (NGFWs) go beyond traditional port and protocol inspection by understanding the applications that generate network traffic. This allows for more refined control over application use within the network, such as allowing certain messaging apps while blocking others.

Firewalls, whether as standalone devices or integrated components of a broader security infrastructure, are indispensable tools in the cybersecurity arsenal. They not only protect networks from external threats but also play a pivotal role in enforcing internal security policies and compliance requirements. As cyber threats evolve, the development and deployment of advanced firewall technologies become increasingly important in safeguarding digital assets and maintaining the integrity of network infrastructures.

Stateful vs Stateless

Stateful and stateless firewalls represent two fundamental approaches to network traffic management and security, each with its own set of advantages and limitations. Understanding the differences between these types of firewalls is crucial for implementing effective network security measures that align with an organization's specific needs and threat landscape.

Stateful Firewalls: Contextual Security

Stateful firewalls offer a dynamic approach to network security by monitoring and maintaining a record of all active network connections. This capability allows them to make informed decisions about which packets to allow or block, based on the complete context of a connection rather than isolated examination of individual packets.

How Stateful Firewalls Work

- **Connection Tracking**: Stateful firewalls track the state of all active connections through a state table, also known as a connection or session table. This table records various attributes of each connection, including IP addresses, port numbers, and the sequence of packets. This information enables the firewall to recognize packets that are part of an existing, authorized connection.
- **Dynamic Filtering**: By maintaining a state table, stateful firewalls can dynamically adjust their filtering decisions based on the flow of traffic. For example, if a user inside the network initiates a web browsing session, the firewall will allow return traffic from the web server by recognizing it as part of an established session.
- **Layer 4 Filtering**: Stateful inspection primarily operates up to the transport layer (Layer 4) of the OSI model, allowing it to understand TCP handshake procedures and UDP communication patterns. This level of inspection helps in identifying and blocking unauthorized attempts to establish a connection.

Advantages of Stateful Firewalls

- **Enhanced Security**: By considering the state of connections, stateful firewalls provide a higher level of security, effectively blocking malicious traffic that does not match the profile of known, safe connections.

- **Intelligent Traffic Management**: They can make more nuanced decisions about traffic, reducing the chances of false positives and negatives in filtering decisions.
- **Efficient Use of Resources**: Stateful firewalls reduce the need for constant rule checking by allowing packets that are part of an established connection, optimizing the use of computational resources.

Stateless Firewalls: Speed and Simplicity

Stateless firewalls, in contrast, adopt a simpler, more straightforward approach to traffic filtering, evaluating packets in isolation without the context of their connection state.

How Stateless Firewalls Work

- **Static Filtering Rules**: Stateless firewalls apply static rules to each packet, inspecting the source and destination IP addresses, port numbers, and the protocol used. These rules are predefined and do not adapt based on ongoing traffic patterns.
- **Rapid Processing**: Without the need to track connection states, stateless firewalls can process packets quickly, leading to potentially higher throughput and lower latency.

Advantages of Stateless Firewalls

- **Speed**: The lack of connection tracking allows for faster packet processing, making stateless firewalls suitable for high-speed environments where latency is a concern.
- **Simplicity**: The straightforward nature of stateless filtering makes these firewalls easier to configure and manage, particularly in scenarios where complex connection tracking is not necessary.

Choosing the Right Firewall

The choice between stateful and stateless firewalls depends on the specific requirements of the network environment, including the need for security versus performance, the types of applications in use, and the overall security posture of the organization. In many cases, a combination of both stateful and stateless filtering may be deployed at different points in a network to balance security and performance needs effectively.

Modern network security strategies often favor stateful firewalls due to their comprehensive security capabilities, especially in environments where the integrity and confidentiality of data are paramount. However, stateless firewalls still find relevance in specific use cases, such as within segmented parts of the network where speed is critical, and the traffic is highly predictable.

In summary, both stateful and stateless firewalls play crucial roles in the landscape of network security. By understanding the capabilities and limitations of each, organizations can better architect their network defenses to protect against the evolving landscape of cyber threats.

How do Firewalls work

Firewalls work by inspecting and regulating the data packets that travel in and out of a network or device, acting as a barrier between a trusted internal network and untrusted external networks, such as the internet. The fundamental purpose of a firewall is to enforce a set of security rules that either allow or block traffic based on predefined security standards. These rules can be highly detailed and are designed to protect against unauthorized access, cyber attacks, and other types of malicious activity. The operation of firewalls can be understood through several key concepts and mechanisms:

Packet Filtering

At the most basic level, firewalls perform packet filtering. Data transmitted over the internet is broken down into smaller units called packets. Each packet contains both the payload (the data itself) and header information, which includes details like the source and destination IP addresses, port numbers, and the protocol being used (e.g., TCP or UDP). Packet filtering firewalls examine the header of each packet against a set of rules and then decide whether to allow it through or block it. This decision is based on the IP addresses, port numbers, and protocols specified in the firewall's rule set.

Stateful and Stateless Inspection

Stateless Inspection is the simpler method where each packet is examined in isolation, without considering the context of the packet within a data stream. Stateless firewalls apply rules based on static values such as source and destination IP addresses, port numbers, and the protocol, making them fast but less secure.

Stateful Inspection, on the other hand, involves tracking the state of active connections (e.g., whether a TCP connection was initiated properly) and making decisions based on the context of the traffic. This method allows the firewall to recognize packets that are part of an established connection, making it more secure than stateless inspection. Stateful firewalls can examine the entire communication path and make more informed decisions about which packets to allow or block, based on the history and state of the connection.

Application-Level Gateway (Proxy Firewall)

Application-level gateways, or proxy firewalls, work by intercepting all incoming and outgoing traffic at the application layer. This type of firewall acts as an intermediary between users and the services they want to access. When a user sends a request to access a service on the internet, the request is first sent to the firewall, which then evaluates the request based on its rules and decides whether to forward it to the destination service. This provides a high level of security because it can control access to specific applications and can inspect the contents of the traffic, looking for malicious data.

Deep Packet Inspection (DPI)

DPI is a sophisticated method that goes beyond examining the packet headers to inspect the actual data within the packets. This allows the firewall to detect and block specific types of content, such as viruses, spam, or certain web applications, based on the payload content. DPI can be used to enforce network policies, such as blocking streaming services or specific websites, and to prevent data exfiltration.

Working Principles

1. **Rule Definition**: Administrators configure firewalls by defining rules that specify which traffic should be allowed or blocked based on IP addresses, domain names, applications, protocols, and ports.
2. **Traffic Inspection**: As data packets arrive at the firewall, it inspects each packet against these predefined rules.
3. **Decision Making**: Depending on the rules, the firewall decides whether to allow the packet to pass through to its destination, block it, or apply other measures such as logging the attempt or sending an alert.
4. **State Tracking** (for stateful firewalls): The firewall maintains a table to track the state of active connections, such as TCP handshakes, to ensure that incoming packets are part of an established connection.

Implementation

Firewalls can be implemented in both hardware and software forms, or as a combination of both. Hardware firewalls are physical devices located between the network and the gateway, providing protection for the entire network. Software firewalls are installed on individual devices, offering tailored protection for that specific device.

In enterprise networks, firewalls often perform deep packet inspection (DPI) to monitor and filter the contents of traffic passing through, ensuring it complies with corporate policies and doesn't pose a security threat. When this traffic is encrypted, as is increasingly common with HTTPS, SSH, and other secure protocols, traditional DPI methods can't inspect the payload directly. To address this, enterprises deploy firewalls capable of SSL/TLS interception or decryption to inspect encrypted traffic. This process involves the firewall acting as a middleman in encrypted communications, decrypting traffic from users, inspecting its contents, then re-encrypting it before sending it to its destination.

Certificate Authority (CA)

For SSL/TLS decryption to work without generating security warnings on users' browsers or clients, the enterprise must deploy its own internal Certificate Authority (CA). This CA is trusted across all devices within the organization, as its certificate is installed on all enterprise-managed devices.

The Process of Decrypting Traffic

1. **Initial Request**: When a user within the network attempts to access an HTTPS-protected site, the firewall intercepts the request. The firewall essentially sits between the user and the internet, acting as the server to the user's client and as the client to the actual server on the internet.
2. **Firewall as a Middleman**: The firewall establishes an SSL/TLS session with the destination server outside the network. It effectively makes the request to the destination server on behalf of the user, initiating a separate encrypted channel with the server.
3. **Decryption and Inspection**: Once the firewall receives the encrypted data from the destination server, it decrypts the data using the server's public key. Now in plain text, the firewall can inspect the contents for malware, data leakage, compliance with corporate policies, and other security threats.
4. **Re-encryption and Delivery**: After inspection, the firewall re-encrypts the data, this time using a certificate issued by the internal CA, and sends it to the user's device. Since the user's device trusts the internal CA, the re-encrypted data is accepted as secure, completing the illusion that a direct, secure connection was established with the original server.
5. **Sending Data Back**: When the user sends data back to the server, the process is reversed. The firewall decrypts the data sent by the user, inspects it, encrypts it again using the destination server's public key, and forwards it.

Security and Privacy Implications

While SSL/TLS decryption allows enterprises to ensure security and compliance by inspecting encrypted traffic, it raises significant privacy concerns and requires careful management of trust. Employees must be informed about the inspection of encrypted traffic, and the private keys used for decryption must be securely managed to prevent abuse.

Additionally, there are technical challenges, such as managing the vast increase in the firewall's workload due to the decryption and re-encryption processes, and the need to keep up with changing encryption standards and protocols to ensure compatibility and security.

In summary, decrypting traffic at the firewall level in enterprise networks allows for thorough security inspection of all encrypted traffic, ensuring malicious content is identified and blocked, and sensitive data is not leaked. However, it must be implemented with careful consideration of privacy concerns, legal requirements, and the technical implications on network performance and security.

Firewall Rules

Firewall rules are the core mechanism through which firewalls enforce security policies on network traffic. These rules determine whether traffic should be allowed or blocked, based on predefined criteria. Firewall rules are essentially instructions or conditions set by network administrators that guide the firewall in making decisions about what to do with each packet of data that attempts to enter or leave the network. Understanding how these rules work requires familiarity with their basic components, how they are structured, and the logic behind their application.

Basic Components of Firewall Rules

1. **Direction**: Specifies whether the rule applies to inbound traffic (entering the network) or outbound traffic (leaving the network).
2. **Action**: Dictates what the firewall should do when the criteria of the rule are met. Common actions include "allow" (permit the traffic), "deny" (block the traffic), and "log" (record the event for review).
3. **Source**: Defines the origin of the traffic. This can be specified by an IP address, a range of IP addresses, or a network zone.
4. **Destination**: Identifies where the traffic is going. Like the source, it can be defined by an IP address, a range, or a network zone.
5. **Protocol**: Specifies the protocol used by the traffic, such as TCP, UDP, ICMP, etc.
6. **Port**: Indicates the port number(s) associated with the traffic. This is particularly important for protocols like TCP and UDP, where services are identified by port numbers (e.g., HTTP on port 80, HTTPS on port 443).

How Firewall Rules Are Structured

Firewall rules are typically processed in a sequential order, from top to bottom. When a packet arrives at the firewall, the firewall starts at the top of the list of rules and checks the packet against each rule in turn until it finds a match. Once a match is found, the corresponding action is taken, and the packet is either allowed to pass, blocked, or logged, depending on the rule's action.

If the packet does not match any rule, a default action is taken, which is typically to deny the traffic to ensure a secure default posture.

Rule Creation and Management

Creating effective firewall rules requires a thorough understanding of the network's architecture, the services that need to be accessible, and potential security threats. Best practices for firewall rule management include:

- **Start with a Default Deny**: Allowing only necessary traffic and denying all other traffic by default is a fundamental security principle. This ensures that only traffic which has been explicitly permitted can access the network.
- **Specificity**: Rules should be as specific as possible to minimize the risk of unauthorized access. For example, if only a specific IP address needs access to a service, the rule should specify that IP address rather than allowing a broader range of IPs.
- **Regular Updates and Reviews**: Firewall rules should be reviewed and updated regularly to ensure they remain effective against evolving threats and aligned with changes in the network configuration or business requirements.
- **Documentation**: Each rule should be documented with information about its purpose, the date it was created, and the person who created it. This documentation is vital for ongoing management and for understanding the rationale behind each rule.

Examples of Firewall Rules

1. Allow inbound HTTP and HTTPS traffic to a web server:
    - Action: Allow
    - Direction: Inbound
    - Source: Any
    - Destination: IP address of the web server
    - Protocol: TCP
    - Port: 80 (HTTP), 443 (HTTPS)
2. Block outbound traffic to a specific IP address:
    - Action: Deny
    - Direction: Outbound
    - Source: Any
    - Destination: Specific IP address
    - Protocol: Any
    - Port: Any

Firewall rules are a powerful tool for securing networks by controlling access and preventing unauthorized traffic. By carefully defining and managing these rules, network administrators can significantly enhance the security of their network environments.

Exercise 1

Windows Server Firewall

ICMP

The Internet Control Message Protocol (ICMP) is a fundamental protocol within the Internet Protocol Suite, as defined by RFC 792. It is used by network devices, including routers, to send error messages and operational information indicating success or failure when communicating with another IP address. For example, if a required service or host isn't available, or if a network device needs to send a packet to inform about a route change, ICMP is used. Although ICMP messages are transmitted within IP packets, ICMP is considered a separate protocol, functioning at the network layer.

Key Features and Uses of ICMP

Error Reporting: ICMP reports errors related to the processing of IP packets. It is essential for diagnosing and reporting network errors, but it does not make any attempt to correct these errors. Common error messages include "Destination Unreachable," "Time Exceeded," and "Parameter Problem."

Diagnostic Functions: ICMP is used for network diagnostics tools such as `ping` and `traceroute`. The `ping` command uses ICMP echo request and echo reply messages to check the health of a connection between two network nodes. `Traceroute` uses ICMP Time Exceeded messages to trace the path from one network device to another, helping to identify the route and measure transit delays.

Network Probing: Administrators can use ICMP to diagnose network performance issues and to map out network topology. By observing how the network responds to various ICMP requests, an administrator can gain insight into the network's behavior.

ICMP DOWN!

Disabling ICMP (Internet Control Message Protocol) on Windows servers is a common security measure taken by network administrators for several reasons. The primary rationale behind this decision revolves around mitigating potential network vulnerabilities and reducing the surface area for attacks. Here are the main reasons why ICMP might be disabled on Windows servers:

1. Preventing Reconnaissance Activities

ICMP can be used by attackers for reconnaissance purposes to discover active hosts on a network by sending echo requests (ping) and waiting for echo replies. By disabling ICMP,

administrators can make their servers less visible to attackers scanning the network, thereby reducing the likelihood of targeted attacks.

2. Mitigating Denial-of-Service (DoS) Attacks

ICMP flood attacks, such as Ping of Death and Smurf attacks, exploit ICMP packets to overwhelm a target server with traffic, leading to denial-of-service conditions. Disabling ICMP can help protect against these types of attacks by preventing the server from responding to ICMP requests, which are often used in the amplification and execution phases of such attacks.

3. Reducing Network Noise

ICMP can generate a significant amount of network "noise" through error messages and informational messages, such as "Destination Unreachable" or "Time Exceeded." In a tightly controlled environment, especially on servers where minimal external communication is necessary, reducing this noise can simplify network management and monitoring.

4. Compliance and Security Policies

Some organizational security policies and compliance standards may require disabling unnecessary services and protocols to minimize potential vulnerabilities. Since ICMP is not required for the essential functioning of a web or application server, it is often disabled to comply with these security best practices.

Implementing the Policy

Disabling ICMP on Windows servers can be achieved through firewall configurations, either by using Windows Firewall or third-party security software. Administrators can create rules to block inbound and outbound ICMP traffic, effectively preventing the server from sending or responding to ICMP messages.

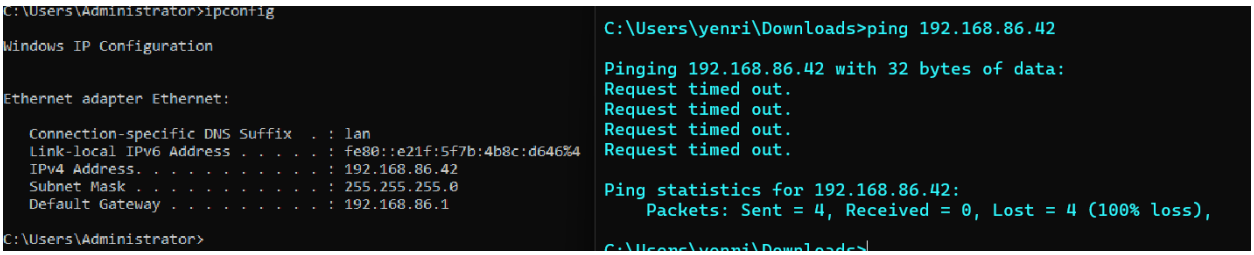
Considerations

While disabling ICMP can enhance security, it may also have drawbacks. For example, it can impede network troubleshooting and monitoring efforts because tools like ping and traceroute rely on ICMP to diagnose network connectivity issues. Therefore, the decision to disable ICMP should be made considering the specific needs of the network environment, balancing security concerns with operational requirements.

In summary, disabling ICMP on Windows servers is a strategic security measure aimed at reducing attack surfaces and protecting against reconnaissance and denial-of-service attacks. However, this measure should be carefully considered and implemented in the context of the organization's overall security posture and operational needs.

Task 1

1. Ensure that your Windows 2019 Sever is in Bridged Virtual Network Mode in VirtualBox
2. Boot your Windows 2019 Server
3. Retrieve your IP Address for your Windows 2019 Server
4. Ping your Windows 2019 Server from your Host Operating system Command Prompt
5. Complete the rest of the following steps



By default Windows has ICMP disabled, so you can't ping it.

I tried a bunch of nmap scans but didn't get much back.

```
C:\Users\yenri\Downloads>nmap -sS 192.168.86.42
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-27 14:57 Central Daylight Time
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 2.18 seconds

C:\Users\yenri\Downloads>nmap -Pn 192.168.86.42
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-27 14:57 Central Daylight Time
Nmap done: 1 IP address (0 hosts up) scanned in 1.63 seconds

C:\Users\yenri\Downloads>nmap -sA 192.168.86.42
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-27 14:58 Central Daylight Time
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 1.64 seconds

C:\Users\yenri\Downloads>nmap 192.168.86.42
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-27 14:59 Central Daylight Time
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 1.64 seconds
```

Finding a Server with ARP Lookup

Even if ICMP is disabled and a server does not respond to ping requests, it's still possible to discover and communicate with it using other methods, such as ARP (Address Resolution Protocol) lookup. ARP is a fundamental protocol used for mapping an IP address to a physical machine address that is recognized in the local network. In simpler terms, when a device on your network wants to communicate with another device on the same network, it uses ARP to translate the recipient's IP address into its MAC (Media Access Control) address. The MAC

address is necessary for the data packets to be correctly routed on the Ethernet layer of the network.

Here's how you can find a server via ARP lookup even if ICMP is disabled:

1. **Direct Communication Attempt**: If you attempt to communicate with the server using its IP address (through a web browser, SSH, or other protocols), your computer will automatically perform an ARP lookup to resolve the server's MAC address. This is because the communication requires the physical address to send packets over the local network.
2. **Checking the ARP Cache**: Once an ARP request is made, the mapping of IP addresses to MAC addresses is stored temporarily in the ARP cache of the requesting device. You can view this cache on most operating systems, including Windows, by using a command-line utility. For example, on Windows, you can open Command Prompt and type `arp -a` to display the current ARP cache, which lists both IP and MAC addresses of devices on the local network.
3. **Passive Scanning**: Tools and network scanners can passively listen to ARP requests and replies on the network to map IP addresses to MAC addresses without actively sending requests. This method can be used to discover devices on a network without directly interacting with them.
4. **Packet Capture**: Packet capture tools, such as Wireshark, tcpdump, or others, can capture and analyze network traffic, providing insights into the data flowing across the network.
    1. **What you'll be looking for**
        1. **ARP Broadcast:** Your computer sends an ARP broadcast request on the network, asking for the MAC address associated with the server's IP address.
        2. **ARP Reply:** If the server is on the local network, it responds with an ARP reply, providing its MAC address.
        3. **Capture Traffic:** Start capturing packets on the network. This is usually done on a computer with network visibility that can see the traffic of interest. In some setups, you might need to configure port mirroring on a switch to direct the traffic to the port where your packet capture tool is listening.
        4. **Filter and Analyze:** Use filters to narrow down the captured traffic to the specific types of communication or protocols you are interested in. For example, you can filter by IP address if you know the server's IP or by protocol if you are looking for specific service communications.
        5. **Identify Server Communication:** Look for patterns in the traffic that indicate server communication. This could include TCP handshakes where the server is the destination, traffic on well-known service ports (like HTTP 80/HTTPS 443), or any other protocol-specific traffic that the server is expected to use.

6. **Extract Server Details:** From the captured and filtered traffic, you can extract details such as the server's IP address, MAC address (from Ethernet headers), ports it is communicating on, and even the nature of the services it is offering or accessing.

It's important to note that while ARP can be used to discover devices on the same local network segment, it does not work across different subnets or through routers. This is because ARP is a link-layer protocol that operates only within the bounds of a single network segment (broadcast domain).

```
C:\Users\yenri\Downloads>arp -a

Interface: 192.168.86.40 --- 0xa
  Internet Address      Physical Address      Type
  192.168.86.1          88-3d-24-95-93-7e     dynamic
  192.168.86.23         08-12-a5-7e-7b-a6     dynamic
  192.168.86.33         10-59-32-ea-63-3b     dynamic
  192.168.86.35         64-cb-e9-f9-6f-64     dynamic
  192.168.86.36         00-f3-61-1d-6e-6b     dynamic
  192.168.86.42         08-00-27-d4-09-a9     dynamic
  192.168.86.43         08-84-9d-bf-b2-64     dynamic
  192.168.86.228        04-d4-c4-04-49-c0     dynamic
  192.168.86.250        08-00-27-df-39-5d     dynamic
  192.168.86.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 192.168.56.1 --- 0x12
  Internet Address      Physical Address      Type
  192.168.56.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

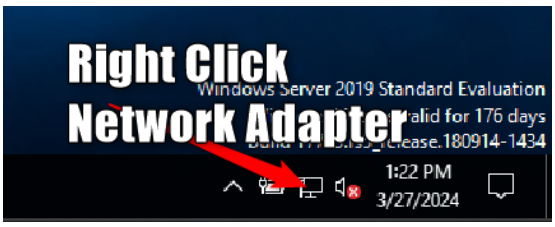
So, just a bit of a recap of previous tools, and techniques that we've previously conducted.

Task 2

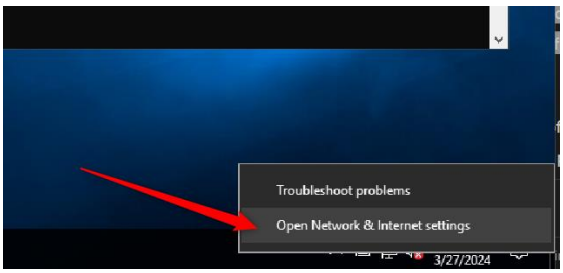
1. Nmap w/ 3 different options
2. Arp and find your server

Task 3

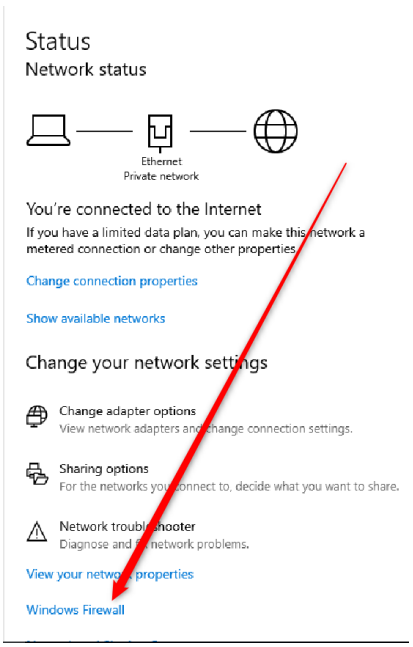
1. Open Windows Firewall



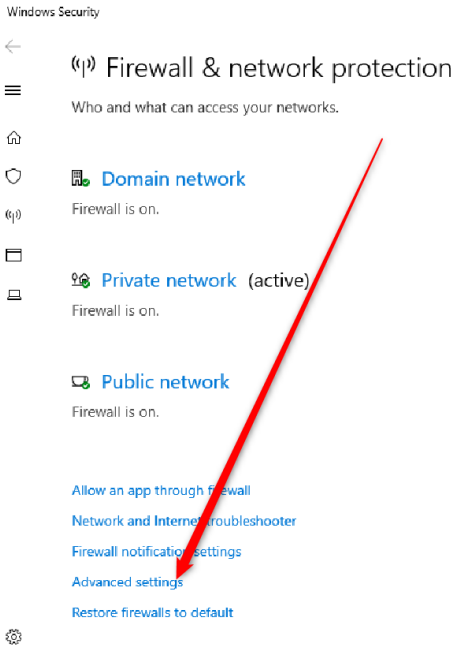
Left Click “Open Network & Internet Settings”



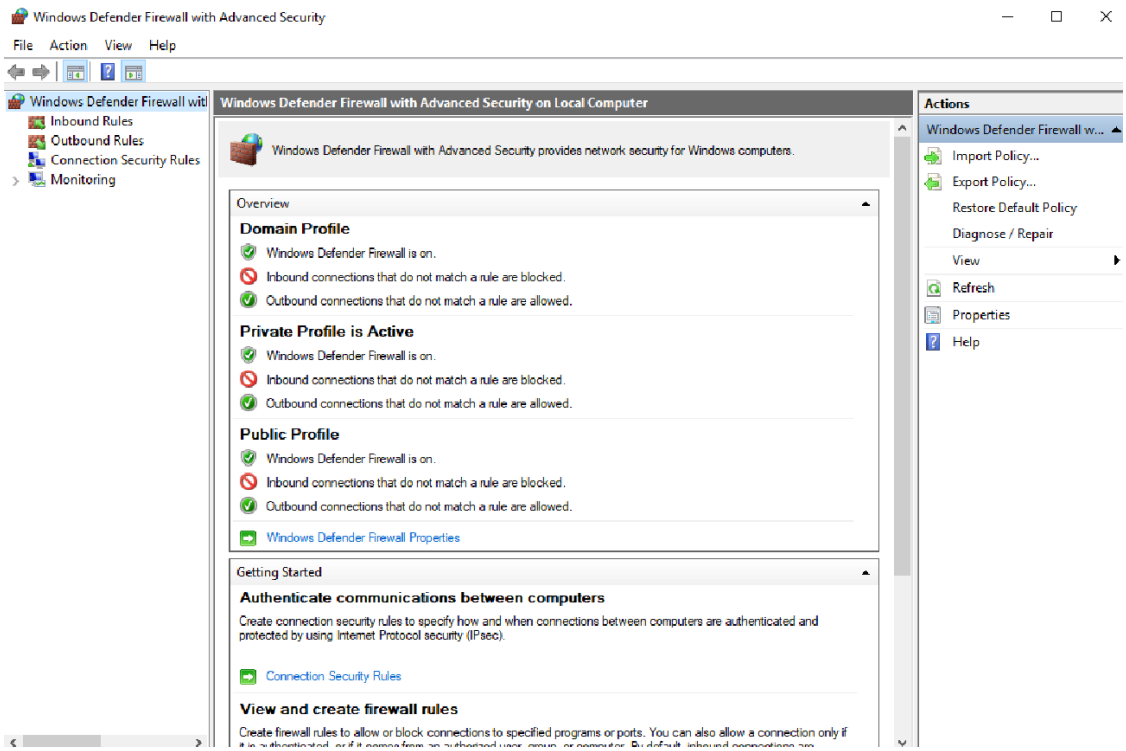
Click on Windows Firewall



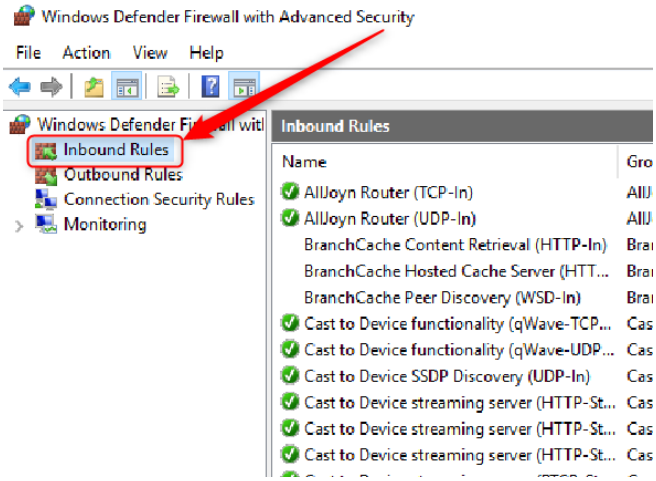
Click on Advanced Settings



Welcome to Windows Firewall



Click on INBOUND RULES



Firewall Rules

Inbound and outbound firewall rules are fundamental concepts in network security, governing how data packets are allowed to enter or leave a network. These rules are essential for controlling access to and from a network, helping to protect against unauthorized access and ensuring that only legitimate traffic is permitted. Understanding the difference between inbound and outbound rules is crucial for configuring firewalls effectively.

Inbound Firewall Rules

Inbound firewall rules, also known as **ingress** rules, dictate the actions a firewall should take with traffic attempting to enter a network. These rules are applied to data packets coming from external sources (outside the network) and trying to access resources within the network.

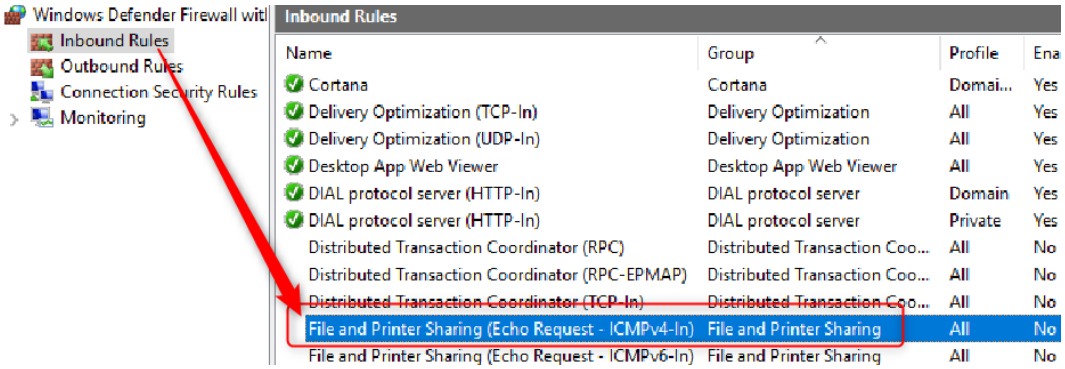
Inbound rules are critical for protecting a network from external threats, such as unauthorized access attempts, malware, and cyber attacks.

Outbound Firewall Rules

Outbound firewall rules, also known as **egress** rules, govern the traffic leaving a network to the external world. These rules are applied to data packets originating from within the network and destined for external locations.

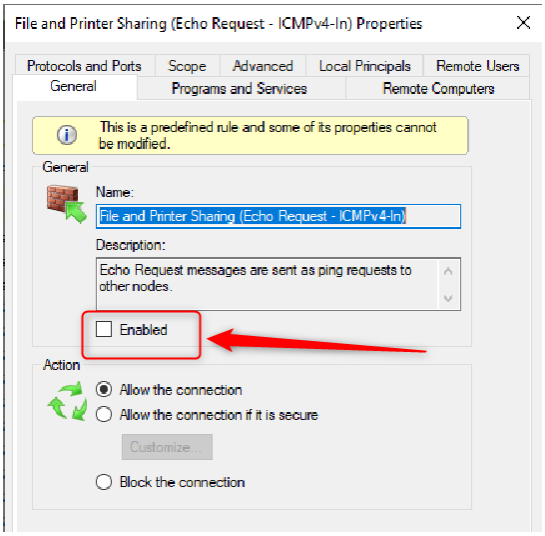
Outbound rules are essential for preventing potentially malicious activities from within the network, controlling data exfiltration, and ensuring compliance with organizational policies regarding the use of external services.

Find “File and Printer Sharing (Echo Request – ICMPv4-In)



2x click on the Firewall RULE

You’ll notice the rule is NOT enabled.



Task 3

1. In your Windows Host OS
    a. Set up a ping stream
        i. ping -t <ip address of your Windows Server>

Example:

```
C:\Users\yenri\Downloads>ping 192.168.68.250 -t

Pinging 192.168.68.250 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
```

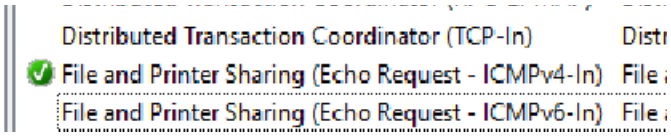
PROTIP – When you are done with the ICMP Stream of Pings, Control-C will break out of the command.

Task 4

1. In your Windows Server Guest OS
    a. ENABLE the Firewall RULE
        i. APPLY and then review your ICMP Steam

Notice that the ICMP Stream immediately starts working

Also look at the firewall rule and the Green icon with the check means that the rule is enabled:



Task 5

1. In your Windows Server Guest OS
    a. Toggle on and off the Firewall RULE
        i. APPLY and then review your ICMP Steam
            1. The stream will start and stop based off your toggling of the firewall rule.

Exercise 2

Windows Server Firewall

IIS – Internet Information Server on Windows

Internet Information Services (IIS) is a flexible, secure, and manageable web server created by Microsoft for use with the Windows NT family. IIS supports HTTP, HTTPS, FTP, FTPS, SMTP, and NNTP. It is an extensible web server software used for hosting websites, web applications, and services. IIS is not just a simple web server but also includes a suite of features that enable developers to deploy and manage web applications and sites easily.

Key Features of IIS

- **Modular Architecture**: IIS has a modular architecture, allowing for the customization of the server by adding or removing modules according to the needs of the application being hosted. This modular approach ensures that the server can be optimized for performance by loading only necessary functionalities.
- **Security**: IIS provides several security features to protect the web server from unauthorized access and attacks. These features include URL authorization, request filtering, and SSL support to encrypt data transmitted over the internet.
- **Management and Administration**: IIS includes a comprehensive set of tools for managing and administering the web server, including the IIS Manager, a graphical interface that provides a user-friendly way to configure IIS and manage websites and applications. PowerShell scripts and command-line tools are also available for automation and scripting tasks.
- **Support for Multiple Protocols**: Beyond just HTTP and HTTPS for web content, IIS supports protocols such as FTP and FTPS for file transfers, SMTP for email services, and NNTP for newsgroups.
- **Application Pool**: IIS uses application pools to isolate web applications, allowing separate configurations and ensuring that issues in one app do not affect others. This isolation improves application security and reduces downtime.
- **Integrated with Windows Authentication**: IIS seamlessly integrates with Windows authentication mechanisms, enabling the use of Active Directory for user authentication. This feature allows for the creation of highly secure environments where access control can be finely tuned.
- **ASP.NET Integration**: IIS is tightly integrated with ASP.NET, a server-side web application framework designed for web development to produce dynamic web pages. This integration allows for efficient hosting and execution of ASP.NET web applications.
- **Scalability**: IIS is designed to scale with your application. It can host anything from small, static websites to large, high-traffic web applications.

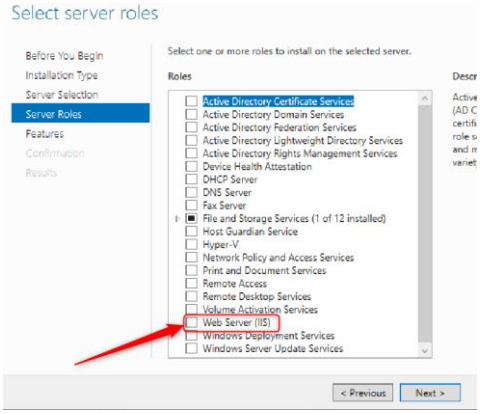
Usage

IIS is used in a variety of scenarios, from hosting simple static websites to complex web applications that require a robust and secure environment. It is particularly favored in environments that utilize other Microsoft technologies, such as ASP.NET for web applications, Windows Server as the operating system, and SQL Server for databases, because of its seamless integration with these products.

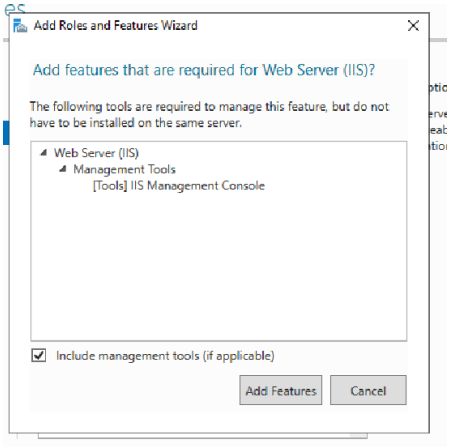
IIS is a critical component for businesses and developers entrenched in the Microsoft ecosystem, providing a powerful platform for deploying and managing web-based applications and services. Its comprehensive feature set and integration with Windows authentication and application development frameworks make it a popular choice for organizations looking for a reliable, scalable, and secure web server solution.

Task 1

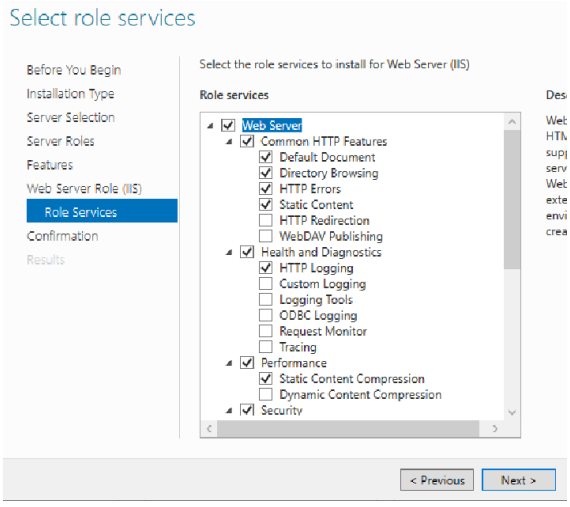
1. In your Windows 2019 Server Guest OS
    a. Run the Server Manager Application
        i. Add Roles or Features
            1. Add IIS
            2.



Go with the default: <click Add Features>



Go with all the default Role Services:

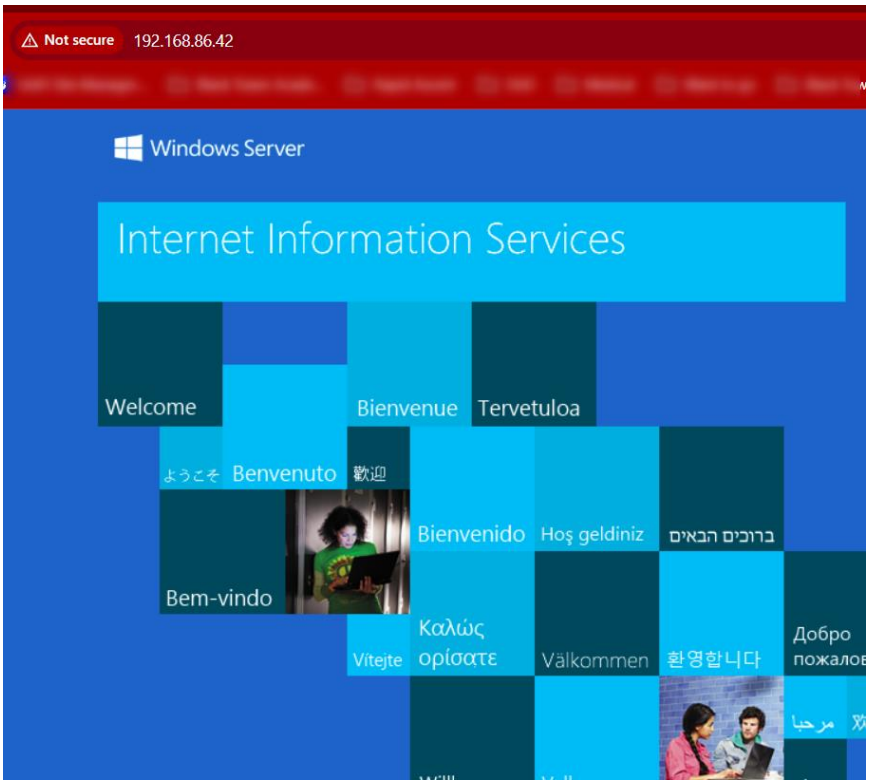


Next till you get the option to install and then install.

=)

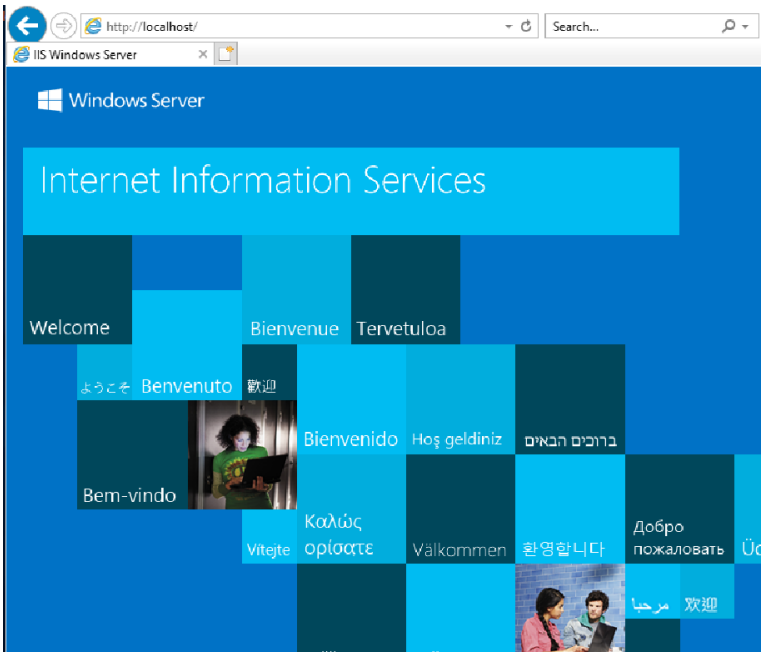
Task 2

1. In your Windows Host OS
    a. Browse to your Windows 2019 Guest OS Web Server
        i. It connects!



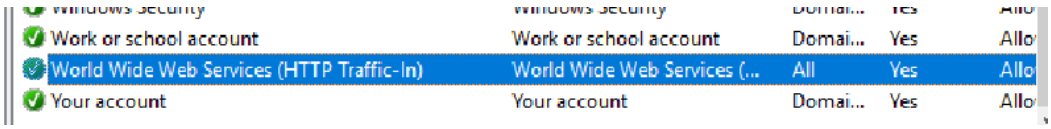
Task 3

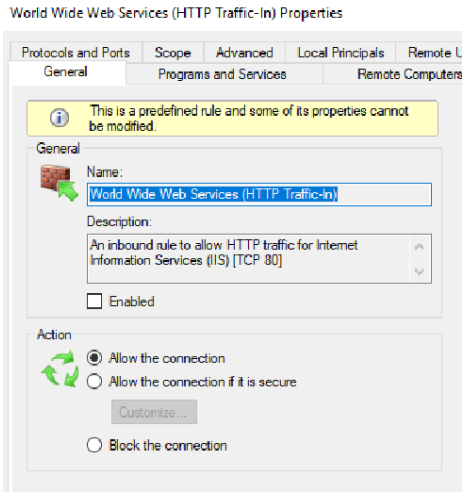
1. From your Windows 2019 Guest OS Web Server
    a. Open the Internet Explorer Browser
        i. Browse to local host
            1. It will show you the default page to the web server



Task 4

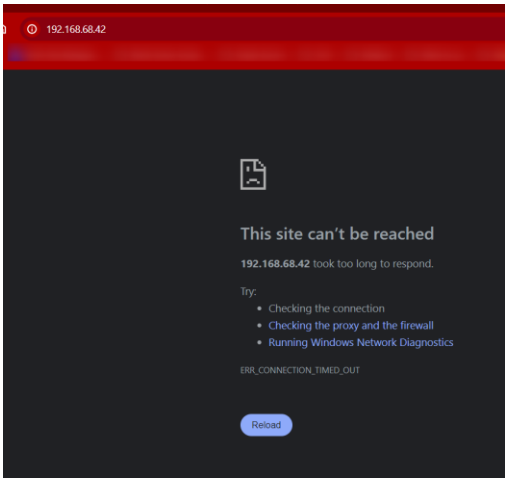
1. From your Windows 2019 Guest OS Server
    a. Open the Advance Firewall View
        i. Find the IIS Firewall Rule
        ii. Disable the IIS Rule





Task 5

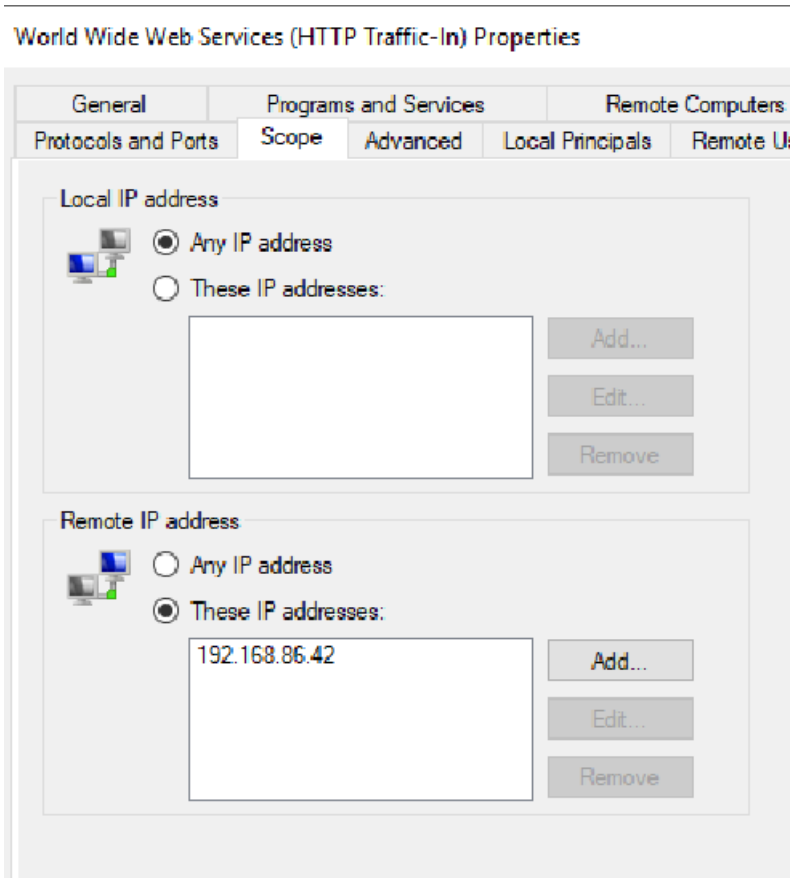
1. In your Windows Host OS
    a. Browse to your Windows 2019 Guest OS Web Server
        i. It fails because of the rule we disabled.



Task 6

1. From your Windows 2019 Guest OS Server
    a. Open the Advance Firewall View
        i. Find the IIS Firewall Rule
        ii. Enable the IIS Rule
        iii. In the SCOPE Tab
            1. Obtain the IP address of your HOST OS
            2. "Remote IP Addresses
                a. Select the "These IP Addresses"
                    i. Insert your HOST OS ip
        iv. Test access to the website from your HOST OS
        v. Find another person's computer, try to navigate to the website on your Windows 2019 Web Server Guest
            1. You will find only your HOST OS can access.

Example:



Exercise 3

Windows Server Firewall

Interacting w/ Windows Firewall via PowerShell

PowerShell can interact with the Windows Firewall through the `NetSecurity` module, which provides a powerful interface for managing Windows Firewall settings, including creating, modifying, and deleting firewall rules. This interaction allows administrators to automate firewall configurations, quickly apply changes across multiple systems, and ensure consistent security policies within a Windows environment. The module offers a comprehensive set of cmdlets that can handle various aspects of the firewall, from basic rule management to more advanced features like configuring security associations and IPsec policies.

Creating Firewall Rules

To create a new firewall rule, the `New-NetFirewallRule` cmdlet is used. This cmdlet allows you to specify various parameters to define the rule's behavior, including its name, direction, action, and the criteria for the traffic it will match.

Listing Firewall Rules

You can view existing firewall rules using the `Get-NetFirewallRule` cmdlet. This cmdlet can be filtered to show only rules that match specific criteria, such as name, enabled status, or direction.

Modifying Firewall Rules

To change the properties of an existing firewall rule, use the `Set-NetFirewallRule` cmdlet. This cmdlet can adjust various aspects of a rule, like its action or the ports it applies to.

Enabling and Disabling Firewall Rules

Firewall rules can be temporarily disabled or re-enabled using the `Enable-NetFirewallRule` and `Disable-NetFirewallRule` cmdlets, without the need to delete or recreate them.

Removing Firewall Rules

When a rule is no longer needed, it can be permanently removed with the `Remove-NetFirewallRule` cmdlet.

PowerShell provides a comprehensive set of tools for managing Windows Firewall rules, allowing for automation and scripting that can significantly streamline the process of securing a Windows network. Through a combination of cmdlets, administrators can create custom rules tailored to their security requirements, manage the active status of these rules, and remove them when they are no longer necessary. This capability is especially valuable in environments where rapid changes to network configurations are common, ensuring that firewall policies can quickly adapt to new security challenges.

Task 1

1. From your Windows 2019 Guest OS Server
    a. Open PowerShell
        i. Run the following command
            1. `Get-NetFirewallRule`
            2. View the output
        ii. Rerun the command but redirect the output to a text file
        iii. Open the text file via Notepad

Creating a Firewall Rule to Block All Traffic from an IP Address

To block all traffic from a specific IP address (for example, 192.168.0.44) using PowerShell, you can use the `New-NetFirewallRule` cmdlet. This cmdlet allows you to specify detailed criteria for the rule, including the action (block or allow), direction (inbound or outbound), and the specific conditions under which the rule should apply (such as the IP address in question).

Here's how to create a firewall rule that blocks all inbound and outbound traffic from the IP address 192.168.0.44:

Task 2

1. From your Windows 2019 Guest OS Server
    a. Open PowerShell
        i. Run the following command
            1. `New-NetFirewallRule -DisplayName "Block Inbound 192.168.0.44" -Direction Inbound -Action Block -RemoteAddress 192.168.0.44`
            2. `New-NetFirewallRule -DisplayName "Block Outbound 192.168.0.44" -Direction Outbound -Action Block -RemoteAddress 192.168.0.44`
            3. View the outputs
            4. Review both Firewall rules in PS using the `Get-NetFirewallRule`
            5. View the outputs
        ii. Find the rules in the GUI version of the Advanced Firewall Rules
            1. What icon is different from the Green Checkmark

a. Why is it different?

iii. Open the text file via Notepad

Explanation of the Cmdlet Parameters:

- `DisplayName`: A name for the firewall rule. This should be something descriptive so you can easily identify the rule later.
- `Direction`: Specifies whether the rule applies to inbound or outbound traffic. In this case, we create two rules, one for each direction.
- `Action`: Defines what action to take when the conditions of the rule are met. Setting this to `Block` prevents the traffic from passing through the firewall.
- `RemoteAddress`: Specifies the IP address or addresses that the rule applies to. In this scenario, it's set to the IP address we want to block.

Managing Firewall Rules with PowerShell

Beyond creating rules, PowerShell allows you to:

- **List existing rules**: `Get-NetFirewallRule` lets you inspect existing rules, filter by name, enabled status, direction, and more.
- **Enable or disable rules**: Use `Enable-NetFirewallRule` and `Disable-NetFirewallRule` to manage the active status of rules without removing them.
- **Modify rules**: `Set-NetFirewallRule` can alter the properties of existing rules, such as changing their action or the addresses they apply to.
- **Remove rules**: `Remove-NetFirewallRule` deletes rules that are no longer needed.

Task 3

1. From your Windows 2019 Guest OS Server
    a. Open PowerShell
        1. Disable both rules
            a. `Block Inbound 192.168.0.44`
            b. `Block Outbound 192.168.0.44`
        2. Re-Enable both rules
            a. `Block Inbound 192.168.0.44`
            b. `Block Outbound 192.168.0.44`
        3.

Task 4

1. From your Windows 2019 Guest OS Server
    a. Open PowerShell
        1. Remove both rules
            a. `Block Inbound 192.168.0.44`
            b. `Block Outbound 192.168.0.44`

Exercise 4

Linux Server Firewall

Interacting w/ Linux Firewall

Ubuntu 22.04, like many Linux distributions, comes with `ufw` (Uncomplicated Firewall) pre-installed as its default firewall management tool. UFW is a user-friendly interface for managing `iptables`, iptables is the traditional firewall utility for Linux that works by setting up rules for allowing or blocking traffic based on the packet's source, destination, and the type of traffic it carries.

UFW aims to simplify firewall management, making it more accessible to users who may not be as familiar with `iptables`' complexity. With UFW, administrators can easily configure, enable, and manage firewall rules through a command-line interface or a graphical user interface (for those who install `gufw`, the GUI version of UFW).

Key Features of UFW in Ubuntu 22.04 Server:

- **Simple Syntax**: UFW allows administrators to manage firewall rules using simple, straightforward commands.
- **Default Policies**: UFW enables setting default policies for inbound and outbound traffic, allowing for a "deny by default" or "allow by default" stance, depending on security needs.
- **Rule Management**: Users can create rules that specify which services and ports are open to traffic. UFW manages these rules in a way that makes it easy to modify or remove them as needed.
- **Logging**: UFW provides logging features, making it easier to monitor attempts to access the server, which can be invaluable for detecting unauthorized access attempts or other suspicious activity.

Managing UFW on Ubuntu 22.04 Server

Here's a brief overview of common UFW commands and how to use them:

- **Enabling and Disabling UFW**:
  - To enable UFW: `sudo ufw enable`
  - To disable UFW: `sudo ufw disable`
- **Setting Default Policies**:
  - To set the default policy to deny all incoming traffic: `sudo ufw default deny incoming`
  - To set the default policy to allow all outgoing traffic: `sudo ufw default allow outgoing`
- **Adding Rules**:

  - To allow traffic on a specific port (e.g., HTTP on port 80): `sudo ufw allow 80`
  - To allow traffic for a specific service by name: `sudo ufw allow http`
  - To allow traffic from a specific IP address: `sudo ufw allow from 192.168.1.1`
- **Deleting Rules**:
  - Rules can be deleted by specifying the same criteria used to create them but replacing `allow` with `delete allow`. For example, to delete a rule allowing HTTP traffic: `sudo ufw delete allow http`
- **Checking the Status of UFW**:
  - To check which rules are currently configured and whether UFW is active: `sudo ufw status verbose`

Considerations for Ubuntu Server Administrators

While UFW simplifies the process of configuring a firewall, it's important for administrators to understand the basics of network security and the implications of the rules they create. Opening too many ports or services can expose the server to unnecessary risk, while overly restrictive rules might hinder legitimate network communication.

Administrators should regularly review firewall logs and rules to ensure they continue to meet the security and operational needs of the Ubuntu 22.04 server. UFW's simplicity and power make it a suitable tool for both novice and experienced users looking to secure their Linux servers.

Task 1

1. Boot up Ubuntu 22.04 Server
    a. Login to the Console
        i. Obtain the IP address of your Linux Server
2. In your Host OS
    a. Open the Windows Command Prompt or System Terminal
    b. `ssh` to your Linux Server
    c. Within the Linux server via `ssh`
        i. `sudo ufw status verbose`
        ii. Observe the output that Status: inactive
3. Enable the UFW Firewall
    a. Your output should be, "Command may disrupt existing ssh connections. Proceed with operation (y|n)?"
    b. `exit` from your SSH session.
4. Attempt to reconnect via `ssh`
    a. Notice that you cannot.
        i. The `ufw` Firewall is now blocking that access because we didn't set up any rules allowing access.

Enabling UFW (Uncomplicated Firewall) without configuring rules for SSH (typically running on port 22) can indeed lock you out of your server when accessing it over the network. This situation occurs because, upon enabling UFW, it starts enforcing its default ruleset. If the default rule is to deny incoming connections and no exception is made for port 22, then SSH connections will be blocked, preventing remote access to the server.

How to Avoid Getting Locked Out

To prevent this from happening, you should explicitly allow SSH connections before enabling UFW. Here's how to do it:

Allow SSH Connections: Before enabling UFW, ensure that you allow SSH connections. You can do this with the following command:

```
sudo ufw allow ssh
```

or

```
sudo ufw allow 22
```

This command creates a rule that allows inbound traffic on port 22, the default port for SSH.

Enable UFW: After configuring the rule for SSH, you can safely enable UFW:

```
sudo ufw enable
```

You'll be asked to confirm the operation. Proceeding will apply the current ruleset, which now includes the rule allowing SSH connections.

Verify the Configuration: It's a good practice to check the status of UFW and review the active rules to ensure that SSH is indeed allowed.

You can do this with:

```
sudo ufw status
```

Additional Tips

- **Check Default Rules**: Before enabling UFW, always check the default policies for incoming and outgoing traffic. By default, UFW is set to deny all incoming connections and allow all outgoing connections. Adjust these settings as needed for your server's security requirements.
- **Use a Physical or Console Access**: If possible, perform firewall configurations while you have physical access to the server or through a remote console access provided by your hosting provider. This way, even if you get locked out due to a misconfiguration, you can still access the server to correct it.
- **Consider Using ufw limit for SSH**: Instead of simply allowing all SSH traffic, you can use the ufw limit ssh command. This limits the rate of incoming connections on the SSH port, helping to protect against brute-force attacks while still allowing legitimate SSH access.

Principle of Least Privilege

The principle of least privilege (PoLP) is a fundamental concept in computer security and IT management, advocating for providing users and programs the minimum levels of access — or permissions — needed to perform their functions. When applied to SSH access on a Linux server, it means granting SSH access only to those users who absolutely need it for their work and ensuring that the permissions and capabilities of those users are restricted to what is necessary for their roles. Here's how to implement the principle of least privilege for SSH access:

Restrict SSH Access to Specific Users

You can limit which users can SSH into the server by configuring the SSH daemon settings. This is done in the `/etc/ssh/sshd_config` file. For example, to allow only users `alice` and `bob` to SSH into the server, you would add or modify the following line:

```
AllowUsers alice bob
```

Alternatively, you can use `AllowGroups` to permit only members of certain groups.

Use SSH Key Authentication

Passwords can be vulnerable to brute-force attacks, and users might choose weak passwords. Instead, use SSH keys for authentication, which are more secure. Require all users who need SSH access to generate an SSH key pair and provide you with their public key, which you then add to their `~/.ssh/authorized_keys` file.

To enforce key authentication, set the following in `/etc/ssh/sshd_config`:

```
PasswordAuthentication no
```

Limit Users' Privileges

For users who need SSH access but do not require full administrative privileges, ensure they do not have `sudo` access or are not part of the `root` group. If they need to perform tasks that require elevated privileges, consider setting up specific `sudo` privileges that allow them to run only the necessary commands as root.

Use Sudo with Logging

For users who must have `sudo` access, configure `/etc/sudoers` using `visudo` to grant only the necessary permissions. Additionally, `sudo` logs all commands executed, providing an audit trail. Ensure that `sudo` configurations are as specific as possible to limit what commands can be run with elevated privileges.

Regularly Review Access

Periodically review who has SSH access to the server and their level of access. Remove SSH access for users who no longer need it and audit `sudo` permissions to ensure they are still in line with the users' current roles.

Implement Two-Factor Authentication (2FA)

For an additional layer of security, consider implementing two-factor authentication for SSH. This requires users to provide not only their SSH key but also a code from a device they own or a message sent to them, adding a second layer of security to the authentication process.

Monitor and Audit SSH Sessions

Use tools to monitor active SSH sessions and audit logs to detect unauthorized access attempts or suspicious activities. Regular auditing helps ensure that the principle of least privilege is maintained and that any deviations are detected and remediated promptly.

Applying the principle of least privilege to SSH access significantly enhances the security posture of your Linux servers. It minimizes the risk of unauthorized access and limits the potential damage that could arise from compromised accounts or insider threats.

Reducing your ssh ATTACK SURFACE

Limiting which IP addresses can SSH (Secure Shell) into a server is a security measure that significantly enhances the protection of the server against unauthorized access and various types of cyber attacks, including brute-force attacks. This approach is often part of a broader security strategy known as "whitelisting," which allows only pre-approved entities access to a particular resource, while blocking all others. Here's how limiting IP addresses improves security:

Reduces the Attack Surface

By only allowing SSH connections from specific, known IP addresses or ranges, you significantly reduce the number of potential attackers who can even attempt to connect to the server. This is akin to reducing the attack surface, which is a fundamental concept in cybersecurity aimed at minimizing the potential entry points for an attack.

Mitigates Brute-Force Attacks

Brute-force attacks involve attempting to login to an account by trying many passwords or passphrase combinations. By restricting SSH access to known IP addresses, unauthorized users from other IPs won't reach the SSH service to even attempt a brute-force attack, thus mitigating this threat.

Prevents Unauthorized Access

If an attacker compromises SSH credentials (e.g., through phishing, malware, or data breach), they still won't be able to access the server via SSH unless their IP address is on the allowed list. This adds an extra layer of security, ensuring that access is not solely dependent on credentials.

Eases Monitoring and Auditing

When SSH access is limited to certain IP addresses, it simplifies the process of monitoring and auditing access attempts. Any login attempt from an unauthorized IP can be immediately flagged for investigation, enhancing the detection of potential security incidents.

How to Implement IP Restrictions for SSH Access

On a Linux server, you can implement IP address restrictions for SSH access using various methods, including:

- **TCP Wrappers**: Use `/etc/hosts.allow` and `/etc/hosts.deny` to specify allowed and denied IPs for SSH. This method is simple but becoming less common in favor of more robust solutions.

- **Firewall Rules**: Configure the server's firewall (e.g., `ufw` in Ubuntu, `firewalld` in CentOS) to only allow incoming SSH connections (`port 22`) from specific IP addresses or ranges.
- **SSH Daemon Configuration**: The SSH daemon itself can be configured to listen for connections only from certain IP addresses, using the `ListenAddress` directive in the `/etc/ssh/sshd_config` file. However, this restricts where the SSH server listens, not the client IPs that can connect.
- **Fail2Ban or Similar Tools**: Use tools like Fail2Ban to monitor SSH login attempts and dynamically block IPs that exhibit suspicious behaviors, such as multiple failed login attempts.

Considerations

While limiting IP addresses for SSH access significantly enhances security, it's essential to maintain an up-to-date list of allowed IPs, especially if the IP addresses of legitimate users are dynamic or change over time. Additionally, consider implementing other security measures such as using SSH keys instead of passwords, enforcing two-factor authentication, and regularly reviewing and auditing SSH access logs.

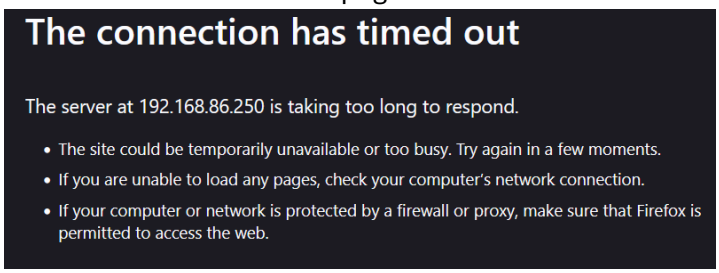
By carefully preparing your UFW configuration before enabling it, you can avoid common pitfalls such as getting locked out of your server. Always ensure that essential services, especially SSH, are allowed in the firewall ruleset to maintain remote access for management and administration tasks.

Task 2

1. Ubuntu 22.04 Server
    a. Login to the Console
        i. Disable the `ufw`
2. In your Host OS
    a. Open the Windows Command Prompt or System Terminal
    b. `ssh` to your Linux Server
    c. Within the Linux server via `ssh`
        i. `sudo ufw status verbose`
        ii. Observe the output that Status: inactive
        iii. Modify the `ufw` to allow ssh traffic
    d. Enable the UFW Firewall
        i. Your output should be, "Command may disrupt existing ssh connections. Proceed with operation (y|n)?"
        ii. `exit` from your SSH session.
    e. Attempt to reconnect via `ssh`
        i. Notice that you can not reconnect since you have set up a rule.

ii. The `ufw` Firewall is now blocking that access because we didn't set up any rules allowing access.

Task 3

1. Ubuntu 22.04 Server
    a. Ensure that apache2 is installed
2. Windows Host
    a. Open a browser and browse to the ip address of your linux server
        i. It will fail to render the webpage
        ii. 

    b. Disable the `ufw` firewall
    c. Refresh your browser and it will immediately render the default web page
        i. 

        ii. This shows that the Software Firewall within the Linux Server is controlling and managing access.
        iii. Toggle the `ufw` firewall on and off proving to yourself that this is in fact the control mechanism.

Firewall's DENY ALL rule concept

The concept of a firewall's "deny all" rule represents a fundamental security posture known as a "default deny" stance. This rule essentially blocks all incoming and outgoing network traffic by default and requires explicit permission for specific types of traffic to be allowed through the firewall.

By implementing a "deny all" rule, a firewall ensures that only traffic which has been specifically authorized, typically through predefined security policies, can access the network. This minimizes the network's exposure to potential threats and unauthorized access, as it prevents any unspecified or unrecognized traffic from entering or leaving the network, thereby **significantly enhancing the security of the system it protects**.

Be Specific

Using specific and more granular configured rules in firewalls significantly enhances security by applying the principle of least privilege to network traffic. This approach ensures that only the traffic necessary for business operations is allowed, while all other traffic is blocked by default. Granular rules can specify which IP addresses, ports, protocols, and applications are allowed or denied, providing precise control over the flow of data into and out of a network.

This granularity reduces the attack surface by limiting potential vectors for unauthorized access or data exfiltration. For instance, if a firewall is configured to allow traffic to a web server, granular rules can restrict access to only the necessary HTTP and HTTPS ports, and even further, only from certain IP ranges that require access. Additionally, specifying allowed applications can prevent malware or unauthorized applications from communicating over the network.

Granular firewall rules also aid in compliance with security policies and regulatory requirements, ensuring that only approved, secure connections are made, and sensitive data is protected. By tailoring the rules to the specific needs of the network, administrators can ensure a tight security posture, monitoring and logging attempts to breach the network, which facilitates the identification and response to potential security threats more efficiently.

In essence, the use of specific and granular firewall rules allows for a more secure, controlled, and compliant network environment, minimizing risks while enabling necessary network functionality and access.

Task 4

1. Ubuntu 22.04 Server
    a. Create a specific rule to allow http traffic
2. Windows Host
    a. Validate the rules effectiveness by being able to access the default page of the Apache Web Server on your Ubuntu Linux Server.

Task 5

1. Ubuntu 22.04 Server
    a. Create a specific rule to allow http traffic only from a range of addresses
    b. Please Note the use of CIDR (You'll have to modify the network for your network)
        i. sudo ufw allow from <your network ID>/<CIDR> to any port 80
        ii. sudo ufw allow from 192.168.1.0/24 to any port 80
2. Windows Host
    a. Validate the rules effectiveness by being able to access the default page of the Apache Web Server on your Ubuntu Linux Server.
    b. Show your `ufw` configuration by using the `status` command.