CST8246 - Basic Internet Services and Automated SSH Configuration

Objectives:

- Establish remote connections to an SSH server as both root and a standard user.
- Manage and restrict SSH access effectively.
- Automate the installation and configuration of OpenSSH using scripts.

Lab Outcomes:

- Properly configure SSH service access.
- Implement firewall rules specific to the SSH service.
- Identify active services and determine the ports they are listening on.
- Explore newly installed packages using utilities like rpm.
- Locate service log files and utilize them for troubleshooting.
- Develop automation scripts to streamline SSH setup and security configurations.

Lab Deliverables:

- Establish SSH access using public key authentication.
- Ensure SSH access is protected by appropriate firewall rules.
- Create and execute an automation script for SSH installation and configuration.
- Detailed demonstration requirements are provided in the Brightspace platform where the lab materials were obtained.

Section A – Configuring SSH

Background Information

This section provides:

- An overview of SSH authentication methods
- An introduction to public-key cryptography
- A brief explanation of the SSH login process using public-key user authentication

SSH Authentication Methods

SSH (Secure Shell) is a protocol designed for secure remote login, incorporating both user authentication and encryption of all communications between client and server. The primary SSH authentication methods include:

1. **Password-Based User Authentication**: Relies on the server's user account database (e.g., password file) for verifying user credentials. This method is controlled by the PasswordAuthentication option in the SSH server configuration.
2. **Host-Based Authentication**: Authenticates SSH clients based on a list of trusted client hosts maintained on the server. This is managed via the HostbasedAuthentication option in the SSH server configuration.
3. **Public-Key User Authentication**: Utilizes public-key cryptography, where the server authenticates users based on their public keys stored on the server. The PubkeyAuthentication option governs this method.
4. **Public-Key Host-Based Authentication**: Similar to host-based authentication but employs public-key cryptography, using a list of client hosts' public keys stored on the server. This is configured with the HostbasedAuthentication option.

In this lab, we will set up two authentication methods:

- **Password-Based User Authentication**: Simple to configure and allows verification of the SSH server's functionality.
- **Public-Key User Authentication**: Offers enhanced security. A primer on public-key cryptography is provided below.

Note: Public-key authentication benefits users by eliminating the need to enter a password at login, as the client and server exchange keys during session setup. However, this requires the user to connect from the same client system or replicate the setup on each client system used to access the server.

Primer on Public-Key Cryptography

Disclaimer: Public-key cryptography can be challenging to grasp initially.

Cryptographic Systems

- The primary goal of encryption is to ensure data confidentiality.
- Two common encryption schemes are:
    - **Symmetric (Shared Key) Cryptography**: Uses a single key for both encryption and decryption. Both sender and receiver must have the key, making secure key exchange a challenge.

- **Asymmetric (Public-Key) Cryptography**: Employs a pair of keys—a public key (publicly available) and a private key (kept secret by the owner). This method addresses the key exchange problem inherent in symmetric cryptography.

Public-Key Cryptography

- Involves a key pair:
  - **Public Key**: Freely distributed and used by others to encrypt messages intended for the key owner.
  - **Private Key**: Kept confidential by the owner and used to decrypt messages encrypted with the corresponding public key.
- The RSA algorithm is one of the most commonly used public-key cryptography algorithms.

SSH User Authentication Using Public-Key Cryptography (Overview)

Session Key (Symmetric Key) Exchange:

1. Upon a client connection, the server sends its public key (host key) to the client.
2. The client verifies the server's authenticity based on the received key.
3. The client generates a session (shared) key and encrypts it with the server’s public key.
4. The server decrypts the session key using its private key.
5. Both client and server now share a session key, enabling symmetric encryption for the duration of the session.

User Authentication Using Public-Key Cryptography:

1. During user login, the server sends a "challenge" (a random string) encrypted with the user's public key.
2. The client decrypts the challenge using the user's private key.
3. The client returns the decrypted challenge to the server, encrypted with the session key.
4. The server decrypts the response using the session key. If it matches the original challenge, authentication is successful.

This process ensures that only users with the correct private key can authenticate, enhancing security by eliminating the need for password transmission.

Secure SSH Configuration Considerations in RHEL 8

When configuring SSH on Red Hat Enterprise Linux (RHEL) 8, there are specific considerations to ensure a secure and compliant setup:

System-Wide Cryptographic Policies:

RHEL 8 introduces system-wide cryptographic policies that govern the security standards for various protocols, including SSH. The default policy, DEFAULT, provides secure settings suitable for current threat models, allowing protocols like TLS 1.2, TLS 1.3, and SSH2. It enforces minimum key sizes, accepting RSA keys and Diffie-Hellman parameters larger than 2047 bits. To modify the cryptographic policy, use the update-crypto-policies command.

SSH Configuration Directory:

In RHEL 8, the SSH daemon (sshd) configuration can be managed using the /etc/ssh/sshd_config.d/ directory. This allows administrators to override default settings by placing custom configuration files in this directory, providing a modular approach to SSH configuration management.

Firewall Configuration for SSH:

To control SSH access, it's essential to configure the firewall appropriately. For example, to allow SSH traffic on a specific interface while restricting it on others, adjust the ListenAddress directive in the /etc/ssh/sshd_config file to bind SSH to the desired IP address. After making changes, restart the SSH service using systemctl restart sshd.

Security Technical Implementation Guides (STIGs):

For environments requiring compliance with Department of Defense (DoD) standards, RHEL 8 provides guidelines to display the Standard Mandatory DoD Notice and Consent Banner before granting SSH access. This involves editing the /etc/ssh/sshd_config file to set the Banner directive pointing to a file containing the required notice.

By incorporating these RHEL 8-specific considerations into your SSH configuration, you can enhance the security and compliance of your systems.

Summary of Key SSH Files

On the Server System:

- **Private Host Key:** /etc/ssh/ssh_host_rsa_key
- **Public Host Key:** /etc/ssh/ssh_host_rsa_key.pub
- **Authorized User Public Keys:** ~/.ssh/authorized_keys

On the Client System:

- **Private User Key:** ~/.ssh/id_rsa
- **Public User Key:** ~/.ssh/id_rsa.pub
- **Known Hosts:** ~/.ssh/known_hosts

These file paths are standard across most Linux distributions, including Red Hat Enterprise Linux 8. Ensuring the correct permissions and configurations for these files is crucial for maintaining secure SSH communications.

SSH Configuration Overview

- **Configuration Files and Service**:
    - Most SSH configuration files for both clients and servers, as well as key files, are located in /etc/ssh/.
    - The SSH server daemon is named sshd.
    - The primary server configuration file is /etc/ssh/sshd_config.
    - The client utility used to connect to an SSH server is ssh.
- **Logging**:
    - To monitor SSH activities, open the relevant log file in a separate terminal and configure it to update continuously.
- **Default Configuration Values**:
    - Default settings apply even if the corresponding entries in the configuration file are commented out.

SSH Installation

On the SSH Server:

- Install or update the necessary OpenSSH packages: openssh-server openssh-clients.

On the SSH Client:

- Install or update the required OpenSSH packages: openssh-clients.

SSH Authentication

Password-Based User Authentication:

- **SSH Service Configuration**:
    - Ensure that password-based authentication is enabled in the SSH server configuration.
        - In the sshd_config file, set the PasswordAuthentication option to yes.
    - Start the SSH service.
        - On the first start, the server generates its host keys. Verify that the RSA key pair has been created in /etc/ssh/.
    - Check the service status to identify the interfaces and ports on which the SSH service is listening.
- **Testing**:
    - As a regular user, log in to your SSH server from the SSH client.
        - During the initial login, you'll be prompted to accept the server's key fingerprint, which will be added to the known_hosts file in the user's home directory on the client system. Verify that the ~/.ssh/ directory on your client system contains the known_hosts file and that it lists the SSH server's fingerprint.
    - Repeat the login process using the root account.
    - For enhanced security, consider disabling root account access via SSH and test to ensure that root login is appropriately restricted.

Public-Key Cryptography-Based User Authentication:

- **Setup on the SSH Server**:
    - Configure SSH to use public-key authentication in addition to password-based authentication.
        - In the sshd_config file, ensure the PubkeyAuthentication option is set to yes.
- **Setup on the SSH Client (as a user)**:
    - Log in as a user and generate a private/public key pair on the client system using ssh-keygen:
        - Execute the command: ssh-keygen -t rsa

        - When prompted for a passphrase, press Enter to proceed without a passphrase (for this setup).
    - Verify that the key files have been created in the ~/.ssh/ directory:
        - The private key is stored in ~/.ssh/id_rsa.
        - The public key is stored in ~/.ssh/id_rsa.pub.
    - Transfer your public key to the authorized_keys file in the ~/.ssh/ directory on the SSH server.
        - The ssh-copy-id script, included with the openssh-clients package, can facilitate this process.
- **Testing**:
    - Attempt to access the SSH server from your client as the user. You should be logged into your home directory without being prompted for a password.
        - Note: If you try to log in as a different user (e.g., root) without the corresponding key setup, you will be prompted for a password.

Section B – IP Aliasing

1. **Create an IP Alias:**
    - On the server, assign an additional IP address to your network interface. This can be achieved by configuring an alias for the network card (NIC). For detailed instructions, refer to Red Hat's guide on assigning alias IP addresses. □cite□turn0search2□
2. **Bind SSH to Specific Interfaces:**
    - Configure the SSH daemon (sshd) to listen on both your existing interface and the newly created alias. This involves specifying the ListenAddress directive in the SSH server configuration file (/etc/ssh/sshd_config). For guidance on binding SSH to specific interfaces, consult Red Hat's solution on configuring sshd to listen on multiple interfaces. □cite□turn0search11□
3. **Test SSH Connectivity:**
    - From the client system, attempt to establish an SSH connection to the server using the new IP alias to ensure proper configuration.

Section C – Securing SSH

Overview: In this section, you'll set up firewall rules to manage access to your SSH server. Specifically, you'll configure the firewall to accept incoming SSH connections from your subnet while blocking SSH traffic from other sources. After implementing and testing these rules, you'll create a script to automate the firewall configuration.

Setup:

Step #1: Configure Firewall Rules

- Utilize firewalld, the default firewall management tool in RHEL 8, to define rules that allow SSH access from your subnet and deny it from others. For comprehensive instructions on using firewalld, refer to Red Hat's documentation on securing networks. □cite□turn0search5□

Step #2: Create a Firewall Script

- **Script Setup:**
  - Develop a script named FWrules.sh in your home directory to automate the application of your firewall rules.
- **Define Variables:**
  - Within the script, set variables as needed to streamline rule definitions.
- **Flush Existing Rules:**
  - Ensure the script clears current firewall rules to start with a clean slate.
- **Set Default Policies:**
  - Configure the default policies for the INPUT, FORWARD, and OUTPUT chains to be permissive.
- **Add SSH Rules:**
  - Implement rules to allow SSH traffic from your subnet (e.g., 172.16.31.0/24) and deny SSH traffic from all other sources.
- **Include Additional Rules:**
  - Add rules to secure any other services or ports that were configured in previous labs.
- **Test the Script:**
  - Execute the script to verify that the firewall rules are applied correctly and that SSH access behaves as intended.
- **Maintain Firewall Script:**

    - Ensure this firewall script is active at all times on your server. It will be evaluated as part of each lab's assessment.

- **Default Rule Considerations:**
    - The default rules should accept traffic from the client subnet (172.16.31.0/24) and block traffic from the server subnet (172.16.30.0/24). Note that this lab and the DNS2 lab may have specific firewall rule requirements.

Sample Script Template:

```
#!/bin/sh

#

# CST8246 – Firewall rules

# <your name>

#

# Variables

<variables>

#

# Flush iptables to start with a clean slate

iptables -F

#

# Set the default policy to be permissive

iptables -P INPUT ACCEPT

iptables -P FORWARD ACCEPT

iptables -P OUTPUT ACCEPT

#

# SSH rule: accept incoming SSH traffic for subnet users
(172.16.31.0/24)

<SSH firewall rule>

#

# SSH rule: deny incoming SSH traffic from all other users

<SSH firewall rule>
```

```
#
# Other rules: deny incoming traffic from all other users
<firewall rule>
```

By following these steps, you'll enhance the security of your SSH service through effective firewall management and ensure that only authorized users within your subnet can access the server.

Exercise – Automating OpenSSH Installation and Configuration on Red Hat 8

Overview:
In this exercise, you will write a script to automate the installation and configuration of OpenSSH on Red Hat 8 servers and clients. The script will set up an alias IP, configure SSH security settings, enable key-based authentication, and restrict root login for enhanced security. By automating these tasks, you will ensure consistency and efficiency in managing SSH access across multiple systems.

1. Install OpenSSH on Server and Client

- Install the required OpenSSH packages on both the server and client systems using the package manager.
- Ensure the SSH service is enabled and running on both machines.
- Refer to Red Hat's documentation on OpenSSH installation for more details.

2. Configure an Alias IP for SSH

- Assign an alias IP to the server's network interface to facilitate SSH connections.
- Update the network configuration files accordingly and restart the network service to apply the changes.

3. Adjust SSH Configuration for Security

- Modify /etc/ssh/sshd_config to enhance security:
    - Disable root login.
    - Allow only key-based authentication.
    - Restrict SSH access to specific users or groups.
- Restart the SSH service to apply the changes.

4. Set Up Key-Based Authentication

- Generate an SSH key pair on the client machine.
- Copy the public key to the server to enable passwordless authentication.

5. Create an Automation Script

- **Script Setup:**
  - Develop a script named setup_ssh.sh in your home directory to automate SSH installation and configuration.
- **Define Variables:**
  - Store the alias IP and other configurable settings in variables for easy modifications.
- **Install OpenSSH:**
  - Use dnf to install OpenSSH packages if they are not already installed.
- **Configure SSH Settings:**
  - Append necessary configurations to /etc/ssh/sshd_config.
- **Enable and Restart Services:**
  - Ensure SSH is enabled at startup and restart the service.
- **Test SSH Connectivity:**
  - Verify the SSH connection from the client using the configured alias IP.

By following these steps, students will automate SSH installation and configuration, ensuring security and efficiency in managing remote connections on Red Hat 8.