**Connecting to a Remote Server in a Private Subnet**

A remote server in a **private subnet** does not have direct internet access or external SSH access from the public internet. Instead, you typically need to use a **jump server** (bastion host) in the public subnet to connect to the private server. The process involves generating SSH keys, copying the public key to the remote server, and ensuring the SSH configurations are correct.

**Steps to Connect to a Remote Server in a Private Subnet**

**1. Generate SSH Key on Source Server (Jump Server)**

To securely connect to the private server, generate an SSH key on the source server (jump server):

**ssh-keygen -t rsa -b 4096**

* **-t rsa**: Specifies the RSA algorithm.
* **-b 4096**: Defines the key length as 4096 bits for added security.
* You will be prompted to save the key in a default location (typically ~/.ssh/id\_rsa), and you can optionally add a passphrase.

This command will generate:

* **Private Key** (id\_rsa/ id\_ed25519): Kept on the source server (do not share this).
* **Public Key** (id\_rsa.pub/ id\_ed25519.pub): This will be copied to the **target server.**

**2. Copy the Public Key to the Target Server**

You want to copy the public key to the private server using the ssh-copy-id command. However, you mentioned that you're receiving a "permission denied" error.

**ssh-copy-id username@remote\_server\_ip**

This error likely occurs because the target server is either blocking SSH access or SSH key-based authentication isn’t properly configured. Some things to consider:

* **SSH access to private servers**: Direct SSH access to a server in a private subnet is usually restricted unless you are going through a jump server or you have a VPN connection to the VPC.

If the ssh-copy-id command doesn’t work directly due to SSH restrictions, you might have to manually copy the public key or use a jump server (bastion host).

**\*\*3. Avoid Editing sshd\_config for Public Key Authentication**

In some cases, you might need to edit the /etc/ssh/sshd\_config file on the target server to allow SSH key authentication, but you'd like to proceed without making those changes.

By default, **public key authentication** should be enabled, but verify these lines **in /etc/ssh/sshd\_config** on the target server:

**PermitRootLogin yes # Keep this disabled unless you need root login**

**PubkeyAuthentication yes**

**PasswordAuthentication yes** # Optional, but you can use password if key-based login fails

If you don’t want to modify these settings, ensure that the keys are set up correctly in the target server's authorized\_keys file.

**\*\*4. Manually Copy the Public Key**

If ssh-copy-id is not working due to permission issues, you can manually copy the public key by following these steps:

1. **On the source server** (where the key was generated):

**cat ~/.ssh/id\_rsa.pub**

Copy the entire public key output.

1. **Log in to the target server** (using existing credentials):

ssh username@remote\_server\_ip

1. **On the target server**, add the public key to the ~/.ssh/authorized\_keys file:

**mkdir -p ~/.ssh**

**echo "your-copied-public-key" >> ~/.ssh/authorized\_keys**

**chmod 700 ~/.ssh**

**chmod 600 ~/.ssh/authorized\_keys**

This manually adds your public key to the authorized\_keys file, allowing key-based SSH access.

**5. Restart SSH Service**

After configuring SSH, if any changes were made (though you're trying to avoid it), restart the SSH service to apply the changes:

**sudo systemctl restart sshd**

**#!/bin/bash**

**# Usage: ./update\_sshd\_config.sh <PubkeyAuthentication> <PasswordAuthentication>**

**# Example: ./update\_sshd\_config.sh yes yes**

**PUBKEY\_AUTH=$1**

**PASSWORD\_AUTH=$2**

**SSHD\_CONFIG="/etc/ssh/sshd\_config"**

**# Check if the script is run as root**

**if [ "$EUID" -ne 0 ]; then**

**echo "Please run as root"**

**exit 1**

**fi**

**# Backup the current sshd\_config file**

**cp $SSHD\_CONFIG "${SSHD\_CONFIG}.bak"**

**# Update PubkeyAuthentication setting**

**if grep -q "^PubkeyAuthentication" $SSHD\_CONFIG; then**

**sed -i "s/^PubkeyAuthentication.\*/PubkeyAuthentication $PUBKEY\_AUTH/" $SSHD\_CONFIG**

**else**

**echo "PubkeyAuthentication $PUBKEY\_AUTH" >> $SSHD\_CONFIG**

**fi**

**# Update PasswordAuthentication setting**

**if grep -q "^PasswordAuthentication" $SSHD\_CONFIG; then**

**sed -i "s/^PasswordAuthentication.\*/PasswordAuthentication $PASSWORD\_AUTH/" $SSHD\_CONFIG**

**else**

**echo "PasswordAuthentication $PASSWORD\_AUTH" >> $SSHD\_CONFIG**

**fi**

**# Restart SSH service to apply changes**

**systemctl restart sshd**

**echo "Updated sshd\_config with PubkeyAuthentication=$PUBKEY\_AUTH and PasswordAuthentication=$PASSWORD\_AUTH"**