

# SSH via PowerShell



## Description:

### SSH via PowerShell

## Scenario:

Many organizations, particularly those with secure environments, prefer not to use third-party tools such as PuTTY or MobaXterm for Secure Shell (SSH) access to their servers. Instead, they can use PowerShell, a built-in Windows tool, to initiate secure SSH connections. PowerShell's native SSH client provides a seamless and secure method for connecting to remote servers.

This guide covers the steps to set up and use PowerShell for SSH access

## Solution/Steps Taken:

### Prerequisites

- **Windows 10/11 or Windows Server 2019/2022** (or later)
- **PowerShell 7** or later (optional, but recommended for enhanced features)
- **OpenSSH Client** feature installed (by default on Windows 10/11)

### 1. Install OpenSSH Client (if not already installed)

OpenSSH Client is usually pre-installed on newer versions of Windows. To verify if it's installed, follow these steps:

1. Open **Settings > Apps > Optional Features**.
2. Scroll down to check if **OpenSSH Client** is listed under **Installed Features**.

### If it's not installed:

1. Click **Add a feature**.
2. Search for **OpenSSH Client**, select it, and click **Install**.

## 2. Open PowerShell

1. Press **Win + X** and select **Windows PowerShell** (or **Windows Terminal**, if you prefer using that).
2. Type `ssh` and press **Enter** to ensure the OpenSSH client is available. If it's installed correctly, you will see SSH command-line options.

## 3. Connecting to a Remote Server via SSH

To initiate an SSH session, use the following command format in PowerShell:

```
ssh username@hostname
```

- **username:** The account name you wish to log in with on the remote server.
- **hostname:** The IP address or domain name of the remote server.

Example:

```
ssh admin@192.168.1.10
```

PowerShell will prompt you to accept the server's host key (if connecting for the first time) and enter your password for the user account.

## 4. Using SSH Key Authentication

For enhanced security, you can use SSH key-based authentication instead of passwords. Here's how to set up key authentication:

### a. Generate SSH Keys (If You Don't Have Them)

To generate an SSH key pair, run the following command in PowerShell:

```
ssh-keygen
```

This will create a public/private key pair, usually stored in `C:\Users\YourUsername\.ssh\`.

### b. Copy Public Key to Remote Server

Use the `ssh-copy-id` command (Linux servers) or manually transfer the public key to the remote server's `~/.ssh/authorized_keys` file:

```
scp C:\Users\YourUsername\.ssh\id_rsa.pub  
username@hostname:/home/username/.ssh/authorized_keys
```

## 5. Advanced SSH Options in PowerShell

- **Specifying a Different Port:** By default, SSH uses port 22. To specify a different port, use the `-p` option:

```
ssh -p 2222 username@hostname
```

- **Verbose Mode:** To troubleshoot connection issues, you can use verbose mode by adding `-v`:

```
ssh -v username@hostname
```

- **Agent Forwarding:** To use SSH agent forwarding (which allows you to use your local SSH keys on remote servers), use the `-A` option:

```
ssh -A username@hostname
```