Lab Exercise 5- Generate and Use SSH Key with Git and GitHub

Pratik Agrawal

500123601

Devops B2

Objective:

To learn how to generate an SSH key, add it to GitHub, and use it to securely connect and push code without repeatedly entering a password.

Prerequisites

- Git installed on your local machine
- GitHub account
- Basic understanding of Git commands

Step 1 – Check for Existing SSH Keys

Run:

```
ls -al ~/.ssh
```

Look for files like id_rsa and id_rsa.pub. If they exist, you may already have an SSH key.

Step 2 – Generate a New SSH Key

Run:

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

- **-t rsa** → key type
- **-b 4096** → key length
- **-C** → comment (your GitHub email)

When prompted:

- Press **Enter** to save in the default location: /home/user/.ssh/id_rsa (Linux/Mac) or C:\Users\<username>\.ssh\id_rsa (Windows)
- Optionally, set a passphrase for extra security.

Step 3 – Start the SSH Agent

```
eval "$(ssh-agent -s)"
```

Step 4 – Add SSH Key to the Agent

```
ssh-add ~/.ssh/id_rsa
```

```
prati@PRATIKs-VIVOBOOK MINGW64 ~/OneDrive/Documents/Devsecops lab/GIT-SSH (master)
$ ls -al ~/.ssh
ls: cannot access '/c/Users/prati/.ssh': No such file or directory

prati@PRATIKs-VIVOBOOK MINGW64 ~/OneDrive/Documents/Devsecops lab/GIT-SSH (master)
$ ssh-keygen -t rsa -b 4096 -C "pratikagrawal787@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/prati/.ssh/id_rsa):
Created directory '/c/Users/prati/.ssh'.
Enter passphrase for "/c/Users/prati/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/prati/.ssh/id_rsa
Your public key has been saved in /c/Users/prati/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:M/1rx+fojNVqxb8VgVi7fMahIigIknsvN8o6zu/kBXo pratikagrawal787@gmail.com
The key's randomart image is:
+---[RSA 4096]----+
|             .   |
| .          o o  |
|o.         . o o |
|... .   ..  . + o|
|. .o . .S... + * |
| .... .  o... o.+|
| ..E+.      ....+|
|o.++..      .=+o+|
|o==+       .o+=+o|
+----[SHA256]-----+
```

```
prati@PRATIKs-VIVOBOOK MINGW64 ~/OneDrive/Documents/Devsecops lab/GIT-SSH (master)
$ eval "$(ssh-agent -s)"
Agent pid 679

prati@PRATIKs-VIVOBOOK MINGW64 ~/OneDrive/Documents/Devsecops lab/GIT-SSH (master)
$ ssh-add ~/.ssh/id_rsa
Enter passphrase for /c/Users/prati/.ssh/id_rsa:
Identity added: /c/Users/prati/.ssh/id_rsa (pratikagrawal787@gmail.com)

prati@PRATIKs-VIVOBOOK MINGW64 ~/OneDrive/Documents/Devsecops lab/GIT-SSH (master)
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCbODNJcFGuxMDWXnda9ilQt2/TcOe82wLNZhSMCua7mKHgjYgzFJpWOoT+u7xKOqQ
ybDBTbyeVyOE8Fa/5HJhKB9iQ5VB2/QH34hyE+GSPP4pLiFpD3nfcJKVhRC9utIG99q2cwYiHI/ceJeG/gOKHE8cOpY/s8MHJhiw/b3
CCmmFiIFygMtBL8LyDQQeLUTh4+7spI/lw1Vflkjs1ETcNSWTS9AAXFAPP69xiRrG884WVKIwTs1WO1svQL4GhkO4pcBvpHB/YpLfav
SDwZlbzaenSxvgpE7VYMijJiV7VQsEnvxoPuWwY/jQgZZd3PxgSH2kzSNwpkc5fxyFQyxuJtyd2Bi69MCU91bdR+3rya2KhkAjU7mRD
5alz/m6EV1vlhXbbs195IdkNwzDczR5QIEQbRbUfVbqDPWXQ6OSYQ14+Lfql5iQER+dIdL9nQGmwCSenqxO7qhpO/HKtCWhUeDP6HWl
AqmJWijGtlz61K1oxBJd79wqdveDPsvCrAXYocbNDqfcA+nW4Ltna0q8GoDXeazVS49VFVYh9XmUY5p8KBawXyhatCwQOgD5ptODKdy
AXotiOlLXMLUp1ilXFOvwRm1U1dqiPmzaiugYbFyw/ZTc/rnwXMm4K8+/KJqCY+5nOQc5ACY44KWO2daUNhUf9sRwHQWPpPgw45oPrN
Q== pratikagrawal787@gmail.com
```

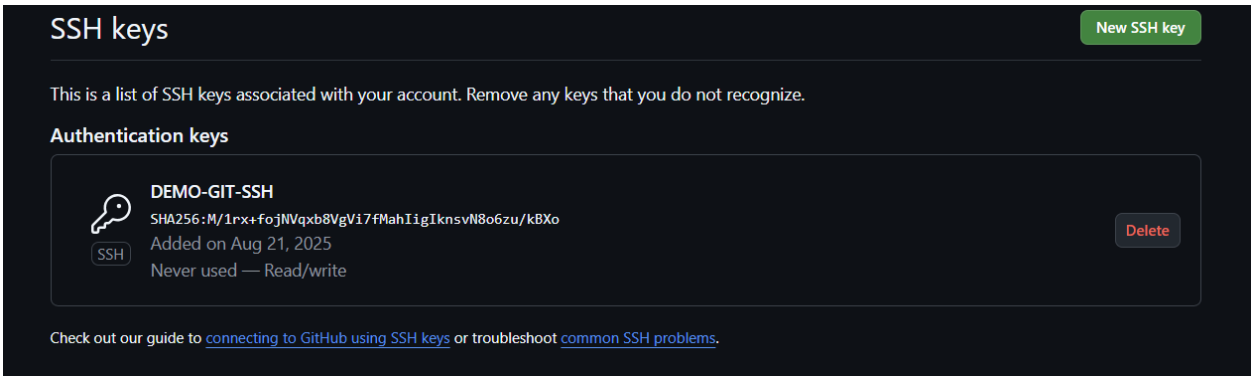
Step 5 – Add SSH Key to GitHub

1. Copy the public key:

```
cat ~/.ssh/id_rsa.pub
```

2. Log in to GitHub → **Settings → SSH and GPG Keys → New SSH key**.

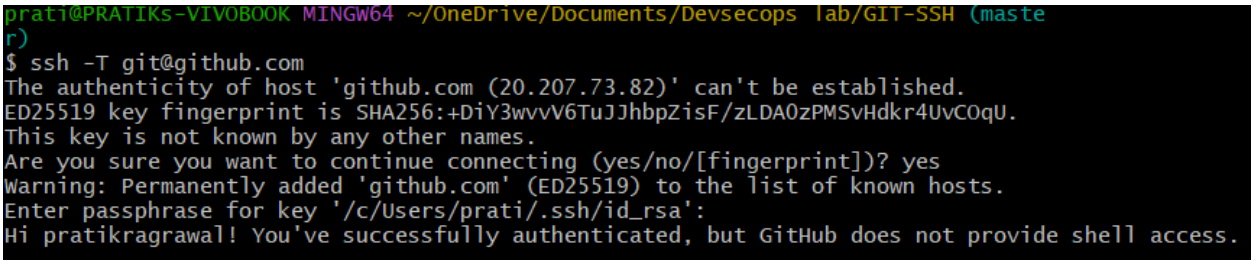
3. Paste the key and save.



Step 6 – Test SSH Connection

```
ssh -T git@github.com
```

Expected output:




```
Hi <username>! You've successfully authenticated, but GitHub does not provide shell access.
```

Step 7 – Use SSH to Clone a Repository

```
git clone git@github.com:<username>/<repository>.git
```

Now you can pull and push without entering your username/password.

```
prati@PRATIKs-VIVOBOOK MINGW64 ~/OneDrive/Documents/Devsecops lab/GIT-SSH (master)
$ git clone https://github.com/pratikragrawal/DEMO-SSH
Cloning into 'DEMO-SSH'...
warning: You appear to have cloned an empty repository.
```

Use Case

Scenario:

An organization's developers often need to push code to GitHub multiple times a day. Using SSH keys eliminates the need to repeatedly enter credentials, while maintaining secure, encrypted communication between the developer's machine and GitHub.

Table – HTTPS vs SSH for GitHub

Feature	**HTTPS**	**SSH**
Authentication	Username & password / token	SSH key pair
Convenience	Requires login each session	No password once key is added
Security	Encrypted, but password-based auth	Encrypted, key-based authentication

Feature	HTTPS	SSH
Best For	Occasional access	Frequent development work