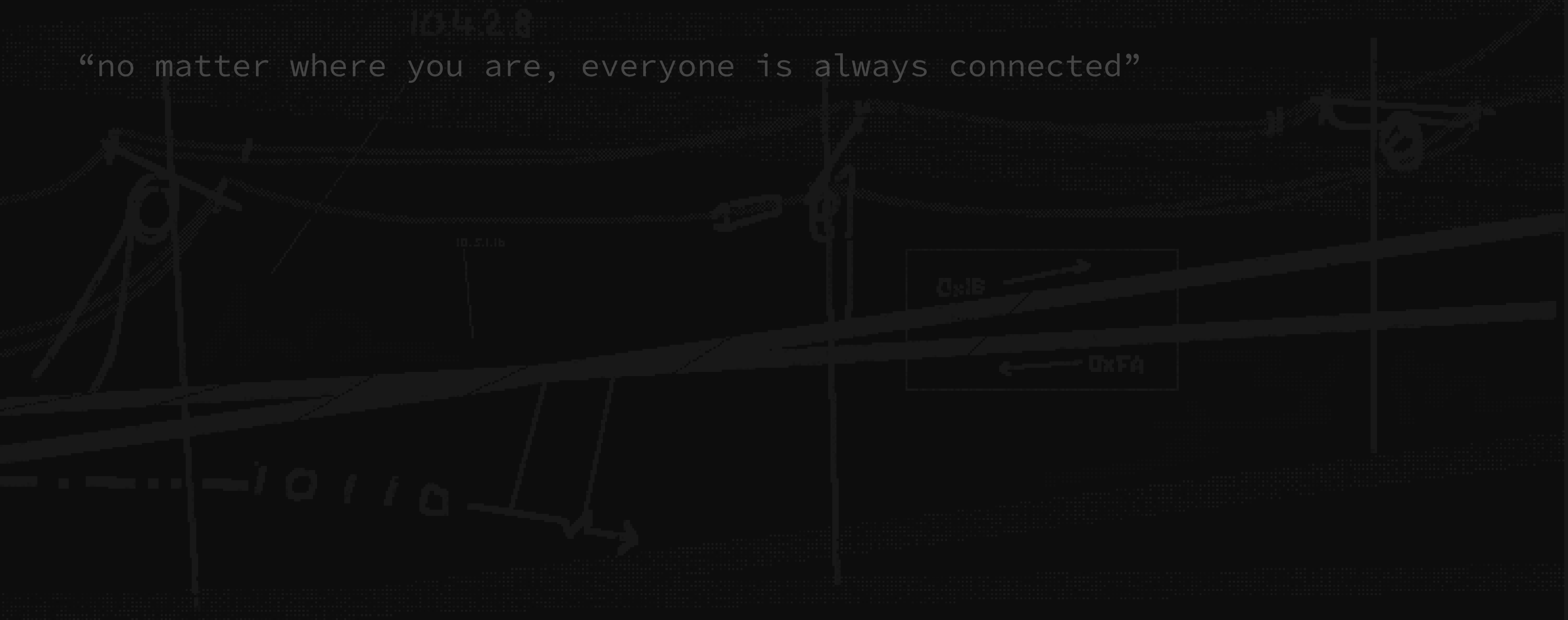


“no matter where you are, everyone is always connected”

11A

SSH with Advanced Config

But I Know SSH…?

Previously, you may have configured SSH with password authentication in mind. We will go over SSH examples with other measures such as key-based authentication.

What You Already (should) Know

in file */etc/ssh/sshd_config*

```
Protocol 2
Port 22
PermitRootLogin no
UsePAM Yes
MaxAuthTries 2
MaxSessions 2
UseDNS no
Banner none
StrictModes yes
X11Forwarding no
X11DisplayOffset 10
```

These following configurations should already be familiar to you.

What You Already (should) Know

in file */etc/ssh/sshd_config*

```
ClientAliveCountMax 0
ClientAliveInterval 300
PrintMotd no
Compression no
LoginGraceTime 30
PrintLastLog no
LogLevel INFO
TCPKeepAlive no
MACs hmac-sha2-256,hmac-sha2-512
```

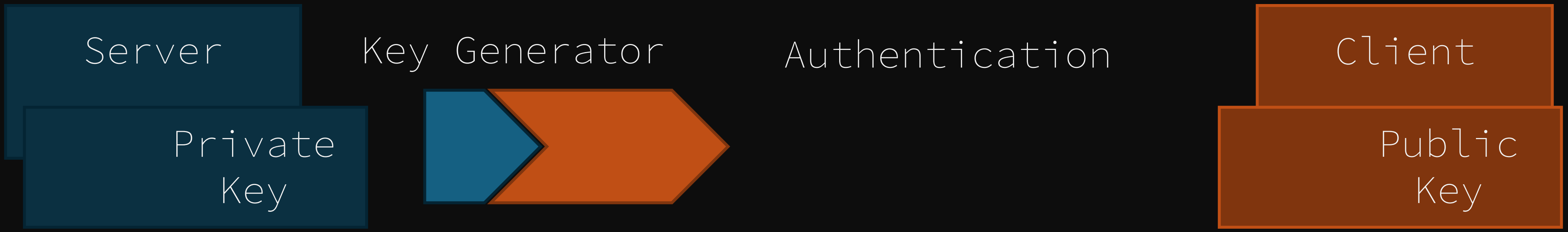
These following configurations should already be familiar to you.

psst… if you don’t have these, write them down!

Introducing Key-Based Authentication

Key-based authentication is an alternative to using passwords to authenticate.

Key-based Authentication



✓ Key Pair Matches

Enabling Key-Based Auth on SSH

in file */etc/ssh/sshd_config*

```
AuthorizedKeysFile
                ~/.ssh/authorized_keys
AuthenticationMethods
                publickey password
```

The private keys for each user is saved in their home directory by default, configured here.

Here, we will allow authentication via public key or password. Other configurations exist.

Enabling Key-Based Auth on SSH

in file */etc/ssh/sshd_config*

```
AuthorizedKeysFile
               ~/.ssh/authorized_keys
AuthenticationMethods
               publickey password
PasswordAuthentication no
```

This will not allow the use of passwords and require public key only.

Remember to *service sshd reload* to apply changes.

Generating a Key Pair

in terminal

```
# su sam
# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/h
Enter passphrase (empty for no passphra
Enter same passphrase again:
Your identification has been saved in /
Your public key has been saved in /home
The key fingerprint is:
SHA256:A1B2C3asdf456DEFjklmetcetc sam@c
The key's randomart image is:
+---[RSA 4096]---+
```

Log in to the user you are generating a key-pair for.

Here, we generate a key using the RSA algorithm with 4096 bits.

This creates two files in ~/.ssh:
- id_rsa (private key)
- id_rsa.pub (public key)

Keys Locked and Loaded

in terminal

```
# ssh-add -l
4096 SHA256:YThyg8vU8RttwAge1lSzHdZATFV
```

Verify that the keys are registered with ssh-agent.

This should spit out the key’s fingerprint as outputted by the previous command.

Authorized Keys Only

in terminal

```
# pwd
/home/sebastian/.ssh
# cat id_rsa.pub > authorized_keys
# cat authorized_keys
ssh-rsa publIcKeyGoesHere123ABC456def+A
```

Add the public key you just generated (~/.ssh/id_rsa.pub) to the authorized_keys file like so

Hostkeys Too!

in file */etc/ssh/sshd_config*

```
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
```

Hostkeys are keys sent by the server upon connection by a client. The client saves this key, and, upon reconnection, checks if the key has changed. If the key has changed, the client may be connecting to a different server, which may indicate malicious activity.

Wrap it Up

in terminal

```
# sshd -T
```

This command validates your configuration. Make sure your configuration is valid before restarting *sshd*! You’ll lose points if *sshd* is down.