UNLV CYBER SECURITY CLUB

Tutorial: PGP Key Creation & SSH Key

Overview

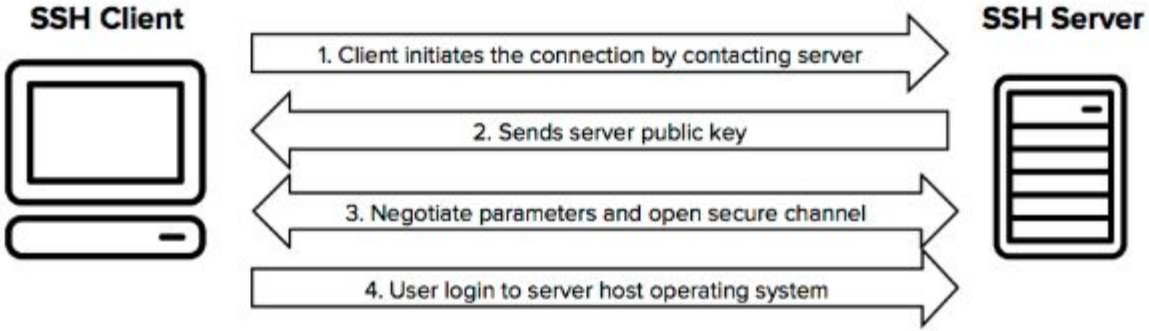
All slides can be found at *https://github.com/layer-zero-unlv/training-sessions*

- History
- Asymmetric Key Encryption
- SSH Key Creation
- PGP Key Creation

History

SSH

- Secure Shell Protocol (Tatu Ylonen 1995)
- Port 22



public

Google Cloud DigitalOcean

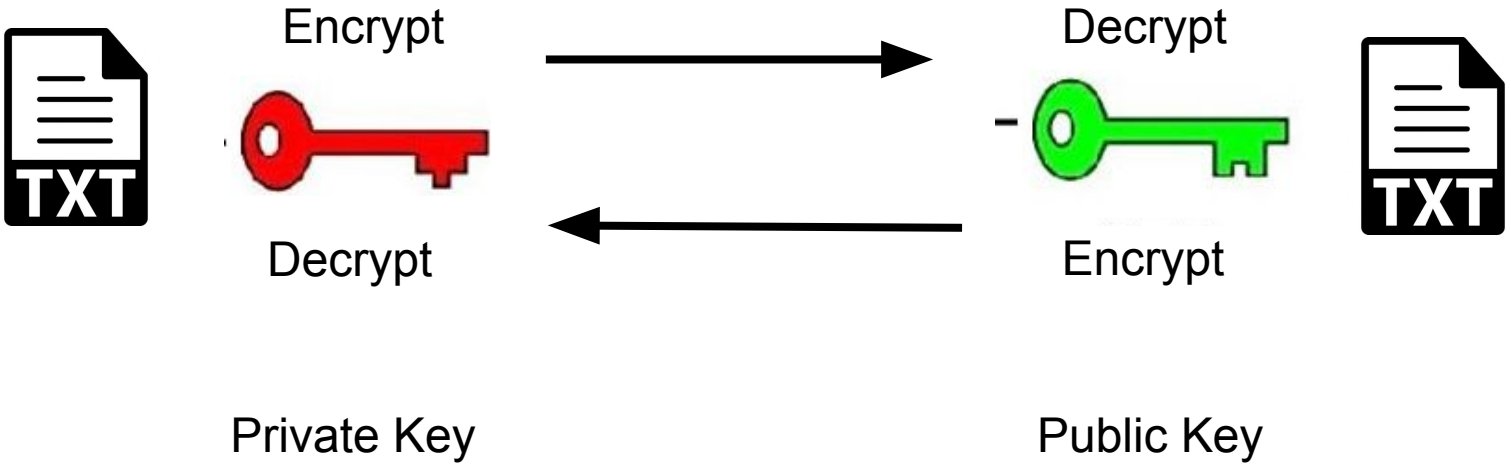
Pretty Good Privacy (PGP)

- Most widely used encryption standard for end-to-end encryption
  - Banks, healthcare, high regulated industries etc....
- Encryption program - Open PGP
  - GnuPG - GNU Implementation
  - Hashing, data compression, symmetric/asymmetric key cryptography
  - Key Management
- End to End Encryption Email
  - Protonmail
  - https://protonmail.com/

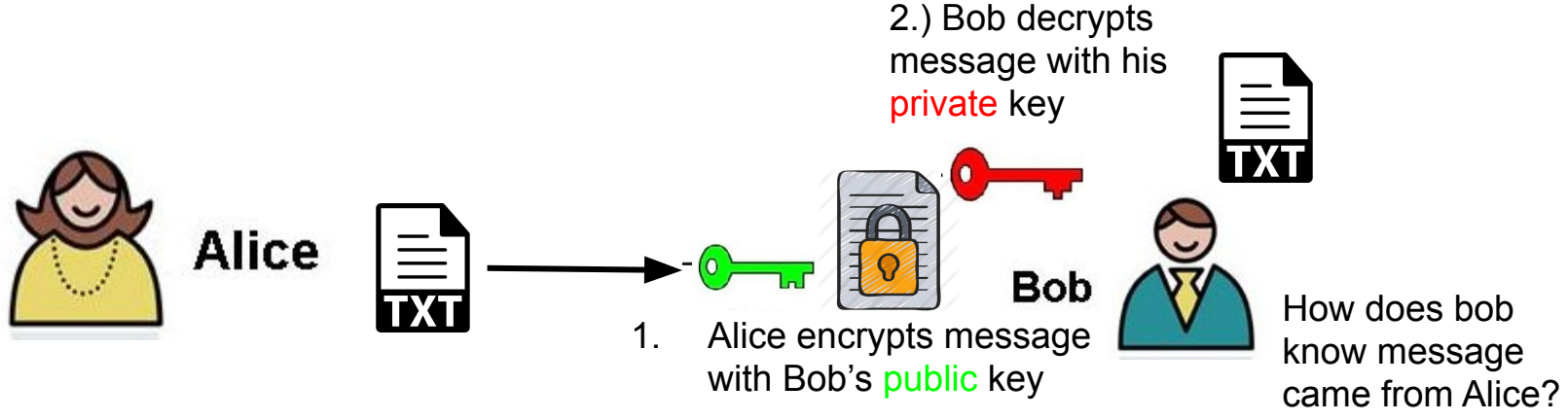
Asymmetric Key Encryption

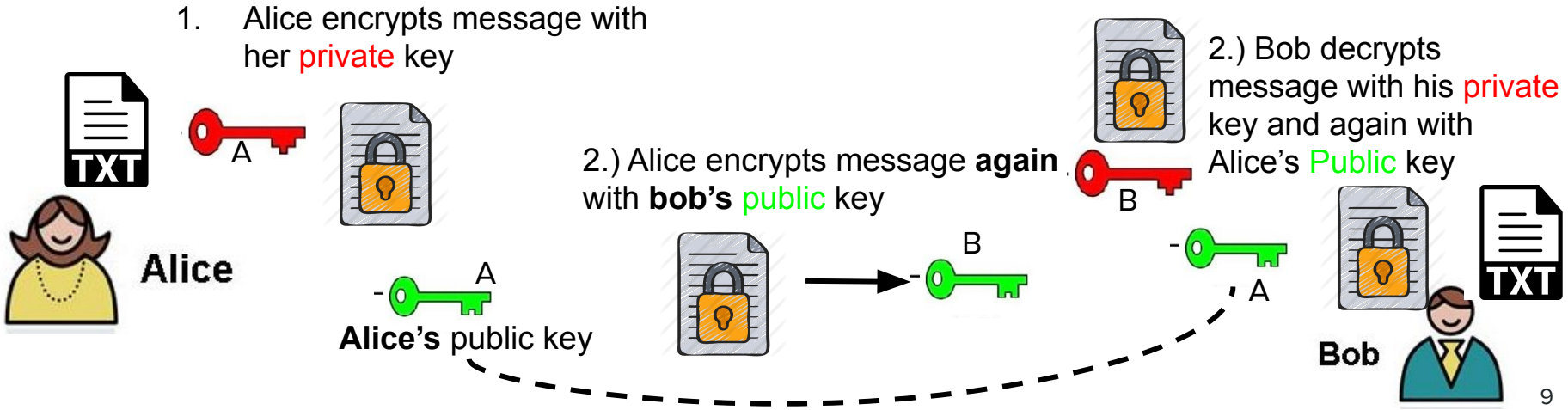
RSA (Rivest–Shamir–Adleman)

- Asymmetric Key Encryption
- Public and Private Keys
- The keys can encrypt and decrypt each other!



RSA


2.) Bob decrypts message with his private key
TXT
Alice
TXT
Bob
1. Alice encrypts message with Bob's public key
How does bob know message came from Alice?


1. Alice encrypts message with her private key
TXT
A
Alice
2.) Alice encrypts message again with bob's public key
B
A
Alice's public key
2.) Bob decrypts message with his private key and again with Alice's Public key
B
A
TXT
Bob

Pretty Good Privacy (PGP)

Gnupg: Create a GPG Key

Check if it's already installed

```
~ $  which gpg
```

If not, run the command to install

```
~ $  sudo apt install gnupg
```

Create your key

```
~ $  gpg --gen-key
```

```
~ $  gpg --full-generate-key
```

GnuPG: Encrypt a Message for Yourself

Create a message

```
~ $  touch message.txt && echo “Hello” > message.txt
```

Encrypt the message to yourself. (use gpg --help to know what the letters are)

```
~ $  gpg -e -a -r <email> <file>
```

Inspect the file

```
~ $  cat message.txt.asc
```

GnuPG: Decrypt the Message From Yourself

Decrypt the message

```
~ $  sudo gpg -o output.txt -d message.txt.asc
```

View the plain text message

```
~ $  cat output.txt
```

GnuPG: Export your Public Key

Export your public key

```
~ $  gpg -a --export <email> > pub-key.asc
```

View your public key

```
~ $  cat pub-key.asc
```

Now email pub-key.asc file to a friend

GnuPG: Import a Public Key

Import key

```
~ $  gpg --import <freinds_key>.asc
```

View your public keys in your key ring

```
~ $  gpg --list-keys
```

GnuPG: Encrypt a message with your friends Key

Encrypt Message

```
~ $  gpg -a -e -r <friends_email> <file>
```

Try decrypting it and see what happens....(it won't work)

Encrypt a message with your key too!

```
~ $  gpg -a -e -r <friends_email> -r <your_email> <file>
```

Decrypt the message sent from your friend again...it works!

```
~ $  gpg -d <file>
```

GnuPG: Decrypt a message

Email the encrypted message to your friend for them to decrypt

Decrypt the message sent from your friend

```
~ $  gpg -d <file>
```

SSH Key Creation

Installing OpenSSH

Check if it’s already installed

```
~ $  sudo systemctl status ssh
```

If not running, need to instal OpenSSH

```
~ $  sudo apt openssh-server
```

Check if the server is running

```
~ $  sudo systemctl status ssh
```

Creating the SSH Key

Ensure ssh-keygen is installed

```
~ $  which ssh-keygen
```

Create the SSH using RSA encryption

```
~ $  ssh-keygen -t rsa
```

Answer the following questions

```
Enter file in which to save the key
(/home/demo/.ssh/id_rsa)
```

```
Enter passphrase (empty for no passphrase):
```

Creating the SSH Key

Ensure key was created

```
~ $  cd ~/demo/.ssh/
```

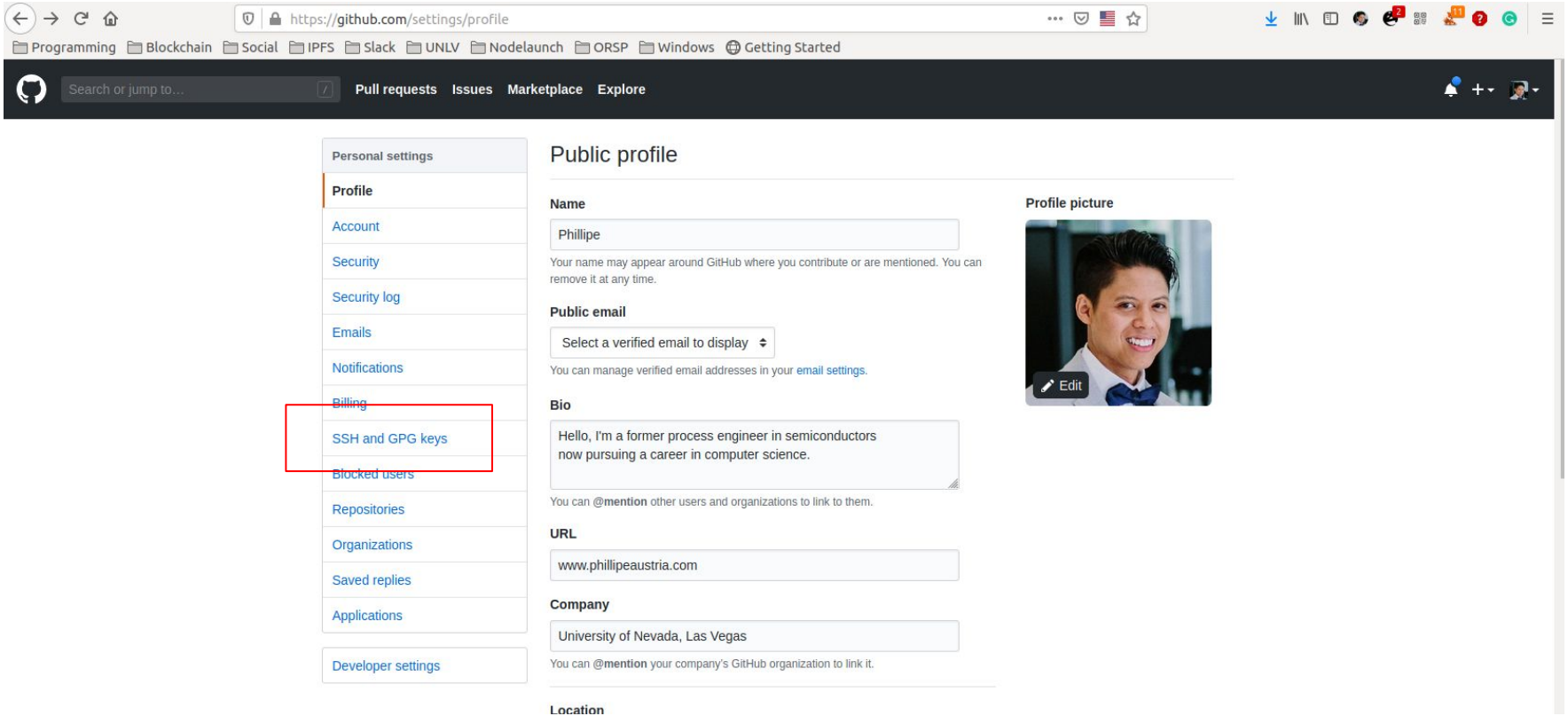
```
~ $  cat id_rsa.pub
```

Answer the following questions

```
Enter file in which to save the key
(/home/demo/.ssh/id_rsa)
```

```
Enter passphrase (empty for no passphrase):
```

Placing it in your Github



Placing it in your Github