CprE 308
Section 3
Lab9

Sean Gordon
Sgordon4

December 10, 2019

These labs focus on an exploration of ssh and the concept of public and private keys. It hilights many parts of secure communication, and demonstrates several commands to manupulate keys and secure files.

I had been meaning to dive more into sshing and to better understand the process, and this lab helped fulfil that. While I was familiar with the concept of public and private keys, I was unaware of many of the commands used with sshing.

3.1.1 Logging in to a remote server:

a) ssh linuxremote1.engineering.iastate.edu

```
The authenticity of host 'linuxremote1.engineering.iastate.edu (10.24.107.153)'
can't be established.
ECDSA key fingerprint is SHA256:hnVVIySw1epHGl6DDP0n5VuWJdQGpyspwbJ/MgoXBSI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'linuxremote1.engineering.iastate.edu,10.24.107.153'
(ECDSA) to the list of known hosts.
sean@linuxremote1.engineering.iastate.edu's password:
```

Output is coming from the remote machine.

b) ssh -l sgordon4 linuxremote1.engineering.iastate.edu

c) ssh sgordon4@linuxremote1.engineering.iastate.edu

3.1.2 Secure file transfer:

a) scp sgordon4@linuxremote1.engineering.iastate.edu: /Desktop/CC.do /Desktop/

b) Idk

3.1.3 SSH escape sequences:

1) ~?
2) scp sgordon4@linuxremote1.engineering.iastate.edu: /Desktop/CC.do ~/Desktop/
3) fg

3.1.4 Known Hosts:

```
|1|Yp1Z/IK/qL6E4qZQGW2aGm93j8E=|TZ4sU2o/DIinOOmXzJ63cKnhbhk=
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdH
AyNTYAAABBBDDb/Bm+amWGwDcVQDw5NCgYx4uMkavihLzx+iKD4s88
bjUXp/gvzPWlVc+oEkCC8maJN/gQpp0C23/ObbGIsjk=
```

3.1.5 Cryptographic keys:

a) id_dsa contains the private key, while id_dsa.pub contains the public one.

b) The passphrase is used to encrypt local keys, preventing unauthorized users from accessing them. If the passphrase is lost you cannot recover it.

c) The passphrase is used to encrypt the private key. The other one shouldn't be encrypted because it is used publicly.

3.1.7 Host authentication and User authentication:

a) On client:

```
> ssh-keygen -t rsa
> ssh-copy-id -i $HOME/.ssh/id_rsa.pub sgordon4linux-1.ece.iastate.edu
> ssh sgordon4@linux-1.ece.iastate.edu
```

b) Entering passphrase to create the key or to use it?
After entering the passphrase to create the key, the key is created and saved in .ssh on local. The public ket is then added to the remote's list of authorized keys. When ssh-ing from local, it will check all of its keys against remote's and use one that matches. This is what happens when, upon attempting to ssh into remote, it prompts for a passphrase rather than a password.

3.1.8 The SSH agent:

a)

```
ssh-agent $SHELL
ssh-add
```

b)

```
ssh ’sgordon4@linux-1.ece.iastate.edu’
Last login: Tue Dec 10 14:20:16 2019 from sgordonlaptop.student.iastate.edu
Setting up environment for: ic assura calibre spectre ext sigrity modus genus incisive innovus
```

3.2.1 Generate a key pair:

```
gpg –gen-key
gpg (GnuPG) 2.2.4; Copyright (C) 2017 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: keybox ’/home/sean/.gnupg/pubring.kbx’ created
Note: Use ”gpg –full-generate-key” for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: SeanGordon
Email address: sgordon4@iastate.edu
You selected this USER-ID:
    ”SeanGordon <sgordon4@iastate.edu>”

Change (N)ame, (E)mail, or (O)kay/(Q)uit? o

We need to generate a lot of random bytes. It is a good idea to perform some other action (type on the keyboard, move the mouse, utilize the disks) during the prime generation; this gives the random number generator a better chance to gain enough entropy.
gpg: /home/sean/.gnupg/trustdb.gpg: trustdb created
```

```
gpg: key 5665C9BFAC7D4870 marked as ultimately trusted
gpg: directory '/home/sean/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/sean/.gnupg/openpgp-revocs.d/B6F043ED078C55A6F
public and secret key created and signed.

pub rsa3072 2019-12-10 [SC] [expires: 2021-12-09]
    B6F043ED078C55A6F09108065665C9BFAC7D4870
uid SeanGordon ¡sgordon4@iastate.edu¿
sub rsa3072 2019-12-10 [E] [expires: 2021-12-09]
```

3.2.2 Exchanging keys:

```
gpg –output rsa3072 –export
gpg –import partnerFile
```

3.2.3 Encrypting and decrypting documents:

a) `gpg –recipient partner –output sgordon4Encrypted –encrypt helloWorld.txt`

b) `gpg –decrypt partnerEncrypted`

3.2.4 Making and verifying signatures:

a) `gpg –output sgordon4Encrypted2 –sign helloWorld.txt`

b) `gpg –decrypt partnerEncrypted2`