DVGC20 VT22 Lab 1: Using Public Keys

Part 1: Encrypted file

Install

1. Install

```
$ wget
"https://github.com/str4d/rage/releases/download/v0.9.1/rage_0.9.1_amd64.deb
"
$ sudo apt install ./rage_0.9.1_amd64.deb -y
```

Create keypair and encrypt

1. Create key-pair

```
$ rage-keygen -o key.txt
Public key: age1rjdtmp2f5zn3ljcd92cfajgtevwtvfnsmxvx7tqmy7lpftwfhs5qwz66rh
```

2. Create file and encrypt using Mahdi's key

```
$ echo "test" > plaintext
$ cat plaintext
test
$ cat plaintext | rage -r
age195v6c8pkyt5erx5nl4malpuuec2gc8z3xewfkylvnan34g6q6u9sdzmckn > crypt.age
```

3. Send email to Mahdi and Samuel with the file *crypt.age* and with the text *only Mahdi can decrypt the attached file.*.

Cleanup

1. Cleanup

```
rm rage_0.9.1_amd64.deb plaintext* crypt.age
sudo apt remove -y rage
```

Part 2: SSH Authentication

1. Connect `ssh oscaande104@hex.cse.kau.se`, got fingerprint `ECDSA key fingerprint is SHA256:pB1lZf5IkBmBfLJXvuycTzHPhaFe6c87VOtsZg7Hl6Q` and selecting *yes* to trust this. Inputing

KAUID password and being greeted with;

```
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-208-generic x86_64)
...
```

2. The connection works, now `exit`.
3. Generate key

```
oscar@DESKTOP-CCRNBR0:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
...
The key's randomart image is:
+--[ED25519 256]--+
|EB.     ...      |
|O+o.   .   .     |
|+*O   .     o .  |
|o*oO o     o + . |
|+.o.*   S . * o  |
| o...    . + . . |
|  .o        . .  |
|  o  o       o   |
|   ++ .     .    |
+----[SHA256]-----+
```

4. Login to the server again using password, and input ssh key fingerprint

```
ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIJiu26bcFkazLEhtFaeGALRQNXLlhzRB5v0Cql5DAFU2
oscar@DESKTOP-CCRNBR0
```

   from `.ssh/id_ed25519.pub` from the client to `~/.ssh/authorized_keys` on the server using vim.
5. `Exit` ssh session.
6. Login using `ssh oscaande104@hex.cse.kau.se -v` to see the debug log, the log says `debug1: Will attempt key: /home/oscar/.ssh/id_ed25519 ED25519 SHA256:d1YLzrcaFACqljDzDPqUwEGIrWZ0AKTg8+s3ugREvAE` and then `debug1: Server accepts key: /home/oscar/.ssh/id_ed25519 ED25519 SHA256:d1YLzrcaFACqljDzDPqUwEGIrWZ0AKTg8+s3ugREvAE` which results in `debug1: Authentication succeeded (publickey).`.
7. Cleanup `rm .ssh/id_ed25519*; truncate .ssh/known_hosts --size 0`

PGP, explain how it works.

PGP (Pretty Good Privacy) is a public-key encryption protocol that estblishes rules for key-pair creation, encryption, decryption and signing. With PGP a user can prove thier identity by signing messages with thier private key as well as encrypting the plaintext message using the recipient's public key.

If a user for security reasons needs to rotate keys, i.e. generate a new keys and prove that thier new key is thier own, they can easiliy do that. Just by signing thier new public key using thier old private key. Altough if the user is doing this becouse thier key is stolen, trust might be breached anyway.

Sources

- https://blog.cryptographyengineering.com/2014/08/13/whats-matter-with-pgp/

What is the difference between PGP, rage and SSH? What protocol do you think is the hardest to use?

PGP is a by now old and also very capable standard for encrypting, signing och decrypting messages using public and private keys. The PGP protocol is implemented by GPG and many others and it's implementation varies between use cases, "in the wild" you will find various setups of algorithms. This leads us into RAGE which is a rust implementation of AGE, AGE is like PGP altough less complex and with fewer options. In the world of secure communication fewer option is good, becouse there is always the psychological aspect of information security, if being secure is too difficult people don't do it right.

SSH is a encryption protocol as well as a transport protocol, it is domain specific and is intended for use with sending commands to Linux and other BSD based systems. Note that SSH is avaliable for Windows as well, but usually protocols like RDP is prefered for that specific platform. SSH has several implementations altough the most popular is OpenSSH.

SSH is widely used and is simple to setup, altough a bit tricky to secure. Best practise is to dissallow passwords and only allow authentication using public keys. As with GPG the key pair generation on the client and the public key matching on the server leaves administrators and users with endless options and choosing the correct algorithm for your use case can be difficult. I find that sticking with `ed25519` is a solid option, if you are trying to connect to a outdated server that has not updated it's server implementation since 2014 you can resort to using the `RSA` algorithm preferably with a bit length no smaller than 2048 bits.

Sources

- https://lwn.net/Articles/583485/

Credit

This document is the work of Oscar Andersson for the course DVGC20 at KAU.