LAB :: SSH 2-Factor Authentication

- In this example we are using `apnictraining.net` as domain name.
- # super user command.
- $ normal user command.
- X replace with your group no.

We will enable two factor authentication for SSH in our Ubuntu server using an OATH-TOTP (open-authentication time-based one-time password) app in addition to an SSH password. We will us Google Authenticator as the PAM (pluggable authentication module).

Step 1: Install Google Authenticator from following link in your Android device/iPhone/iPad/BlackBerry/Firefox devices:

```
https://support.google.com/accounts/answer/1066447?hl=en
```

Follow the instruction and install the app in your mobile.

Or you can search for `google-autheticator` in Google Play or Apple Store.

Step 2: Create an Authentication Key

Install `google-authenticator` PAM first:

```
sudo apt-get update

sudo apt-get install libpam-google-authenticator
```

Log in as the user you'll be logging in with remotely (`apnic`) and run the google-authenticator command to create a secret key for that user.

```
$ google-authenticator

Do you want authentication tokens to be time-based (y/n) y
```

You will get some QR code ouput like bellow:

You will be prompted for some configurations.

1. Scan the QRcode that appears with the Google Authenticator app or you can add the secret key to Google Authenticator app.
2. Save the backup codes listed somewhere safe. They will allow you to regain access if you lose your phone with the Authenticator app.
3. Next it will ask several question; unless you have a good reason to, the defaults presented are sane. Just enter "y" for them.

```
Do you want me to update your "/home/tashi/.google_authenticator" file (y/n)

Do you want to disallow multiple uses of the same authentication
token? This restricts you to one login about every 30s, but it increases
your chances to notice or even prevent man-in-the-middle attacks (y/n)

By default, tokens are good for 30 seconds and in order to compensate for
possible time-skew between the client and the server, we allow an extra
token before and after the current time. If you experience problems with poor
time synchronization, you can increase the window from its default
size of 1:30min to about 4min. Do you want to do so (y/n)

If the computer that you are logging into isn't hardened against brute-force
login attempts, you can enable rate-limiting for the authentication module.
By default, this limits attackers to no more than 3 login attempts every 30s.
Do you want to enable rate-limiting (y/n)
```

Step 3: Now enable google-autheticator in SSH

We will edit `/etc/pam.d/ssh`, that describes which PAM (pluggable authentication module) modules take care of SSH authentication:

```
sudo vi /etc/pam.d/sshd
```

Add the following line (which forces this authentication module to be 'required' and not 'optional'):

```
auth required pam_google_authenticator.so
```

`:wq` Save and quit.

Then edit the SSH configuration file to support 2-factor authentication:

```
sudo vi /etc/ssh/sshd_config
```

Search for `ChallengeResponseAuthentication` and replace `no` with `yes`

```
ChallengeResponseAuthentication yes
```

`:wq` Save and quit.

Now you need to reload the ssh service. You can do it to way:

```
sudo service ssh restart
```

or

```
ps -ef | grep ssh
kill -HUP <process id>
```

replace with correspondence process id.

Step 4: Login to the server

Try to ssh to the server from a new terminal. It will ask for the verification code.

```
                    ***END OF EXERCISE***
```