Passwordless Authentication

Using asymmetric cryptography without having any idea what it means

---

Here's a short list of things that annoy me in life:

- constantly typing the same thing, like a password
- Knowing I have to write my SAGES portfolio to graduate some day

While this guide won't help you write a paper, it will help you use a more secure and easier form of authentication for `SSH`ing into a server.

What is asymmetric cryptography?

you generate a "keypair", which has two parts: public and private. You give someone the public key (pubkey) and tell them "that key is me", but you keep the private key to prove it's you. Similar to a PO box: anyone can put something into your mailbox using only your address (pubkey), but only you can open the mailbox with your key (private key) and read the mail.

Generating your keypair

On your local OSX/Linux computer, run the following command:

```
ssh-keygen
```

The three questions can be skipped by pressing the `return` key, and will be given these default values:
Yes, you want your keypair stored `~/.ssh`.
No, you don't want to put a password on your key.
No, you still don't want to put a password on your key.

While putting a password on your key is more secure, that means you still need to type a password every time you use this key. This is a matter of personal preference.

After completing this command, two new files will be created in your home directory: `~/.ssh/id_rsa`, which is your private key, and `~/.ssh/id_rsa.pub`, which is your public key. You should never give anyone your private key, and shouldn't need to copy it anywhere unless you're backing it up.

Copying your public key to the server

Using the handy `scp` command, you can copy your pubkey over to the server:

```
scp ~/.ssh/id_rsa.pub abc123@eecslinab3.case.edu:~
```

Making the server use your public key

Now you have the pubkey sitting on the server, but the server doesn't know to use it. To tell the server to use it, you need to put the key in a specific file. First you must log into the server:

```
ssh abc123@eecslinab3.case.edu
```

Then you can move your pubkey to the proper location, and clean up:

```
mkdir -p ~/.ssh
cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
chmod -R 700 ~/.ssh
rm ~/id_rsa.pub
exit
```

This string of commands will make the `~/.ssh` directory if it doesn't exist,
copy your pubkey to the proper location. If the file already exists, it's properly appended.
ensure only you can read and write to this proper location
remove the temporary copy of your pubkey
and log out of the server.

You can verify everything was successful by `SSH`ing into the server again, and seeing that it doesn't ask for your password anymore. This will only work on the computer you set it up on.