**SSH Private Keys and Hardening**

**It’s possible to use public key authentication over passwords to log in to SSH**

* More secure than username and password

In order to do this, you must generate a public and private key and put the public on the SSH server

ssh-keygen -t -ed25519 -C “tim@debian”

* Generates key pair
* -ed25519 is the algorithm
* Tim is the user it’s for
* Debian is the client that its coming from

ssh-keygen -t rsa -b 4096 -C “tim@mac”

* Mac code using RSA

Following execution of the code, the key pair will be generated and saved to a directory

You can cat the files to view the keys

Keys will then need to be copied onto the SSH server, within the Tim directory so that when he connects to the server, it knows to look for the key in his directories and he can log in without a password

Cat the public key, pipe it to the SSH server and make a (.ssh) directory and then cat the pub key into the .ssh/Authorized\_keys dir in Tim’s homedir

Cat ~/.ssh/id\_ed25519.pub │ ssh [tim@demo.stationx.net](mailto:tim@demo.stationx.net) “mkdir -p ~/.ssh cat && cat >> ~/.ssh/Authorized\_keys”

Only use this^ (manual code) when the automatic code isn’t available:

* ssh-copy-id [time@demo.stationx.net](mailto:time@demo.stationx.net)

**SSH Hardening**

Wiki.mozilla.org/security/guidelines/openssh

Updated guidelines on what configurations need to be made to the sshd\_config file

* key algorithms
* ciphers
* MACs
* Password-based/ key-based logins
* Verbose logging
  + Uses fingerprints on login
* Log sftp file access
  + To log activity on file transfers
* Prevent root user from being able to login
  + Allow su and sudo commands
* Using kernel sandbox mechanisms

2-FA can also be setup in the config file

Can also use your GnuPG or OpenPGP keys as your SSH authentication

* Incenp.org/notes/2014/gnupg-for-ssh-authentication.html