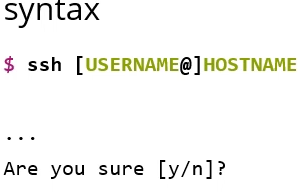
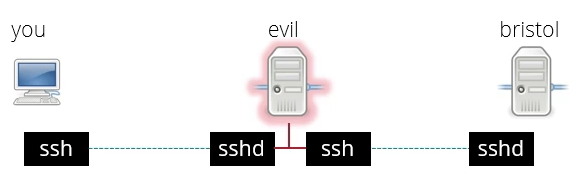
**SSH**

* Secure Shell
* When working with a local system, you have your hardware that connects with input and output like your hardware and screen, which run programs where you type something in & all running on the same machine
* SSH is a pair of programs that allow you to run programs on a different machine as if it were your own
* SSH connects to a server (daemon) SSHD running on another machine
* The connection between the two is encrypted/authenticated in various different ways
* When you type something on your keyboard, SSH encrypts it, sends it to SSHD which encrypts it again, sends to OS on that machine, and shell (or whatever) executes it, encrypts it, sends it back, and it arrives on your screen
* The command to run SSH:



* If username isn’t the same as on your host machine, provide that followed by @HOSTNAME
* SSH is trying to protect you from something called “man in the middle attack” which steals your password



* “evil” server pretending to be Bristol that can decrypt the information, looks like you and Bristol are connected but actually connected to evil server that can keep a record of everything you do
* To protect against this, every server when it’s installed has to have a key pair for digital signatures (a private and public key)
* When you connect to a new server, it says “do you trust this server” and gives you a public key so you can see if connecting to real Bristol server or not
* If you connect to server and key has changed, you get a warning – shouldn’t happen, email IT services
* If you want to connect to a server like GitHub, you have to create your own key
  + Convenient in that you don’t have to be constantly typing your university password every time you want to connect to a server – can use keys
  + Secure – because you’re no longer using your password, you don’t have to enter it, keys are longer so difficult to guess, and keys never leave the machine even if you connect to an evil server
  + When you connect to a server, you have a signature key
  + The server will send you a challenge message
  + (SSH will do this automatically) on your machine, you have to sign the message the server sends you and SSH will send this signature back
  + Just because the server has one signature on your key doesn’t mean it can forge your signature for other messages
* Details of how to do this in practice are in the exercises

