



Intelligent  
Systems  
Lab

---

# Administration of Operating Systems

SSH

Chapter 18

November 7, 2011

# SSH



- *Secure Shell*
  - access remote systems
  - encrypting all traffic
  - host verification
- Uses public key encryption
- ssh client is installed by default on virtually every GNU/Linux systems
- ssh server can be installed using `sudo apt-get install openssh-server`
- ssh client for Windows
  - putty
- Protocol version 2

# Tools

---



Intelligent  
Systems  
Lab

- ssh
  - remote shell
- sshd
  - server
- scp
  - file copy
- sftp
  - file transfer protocol
- ssh-keygen
  - private/public key generation
- ssh-agent
  - secure passphrase storage

# Tools

---



- `ssh host`
- `ssh user@host`
- `exit`
- `ssh user@host command`
- `scp source user@host:destination`
- `scp user@host:source destination`
- `sudo /etc/init.d/ssh restart`
- `sudo /etc/init.d/ssh reload`
- `ssh-keygen -t rsa -f filename`

# Basic Security

---



- Make sure users use secure passwords
  - this is *way* more important than all the rest
- Use public key based authentication
  - more about it later
- Disable remote login for root
  - limit users who have ssh access
- Chroot sshd
  - lock down users to their home directories
- Thwart ssh brute force attacks
- Change the default port 22
- Only use SSH protocol 2

# Symmetric Encryption

---



- Alice and Bob both know the key
  - share secret knowledge
- The key is unknown to Eve
- Problem: how to distribute encryption keys?
  - ... securely ...
  - ... among peers who never interacted before...
  - ... and who can communicate only through a network which is inherently non-secure
- Historically, keys were agreed upon personally
  - meet once, then you can communicate
- Does not work well in the global Internet
  - can be done, but it's quite inefficient

# Establishing a Secret

---



- Alice sends Bob a number of unique puzzles
  - each puzzle has a name
  - there is *a lot* of them
- Bob chooses one at random
- Bob solves the chosen puzzle
- Bob sends Alice the solution
- Now, both Bob and Alice know the secret
  - name of the puzzle Bob has chosen
- Eve does not know the name
  - unless she solved *all* the puzzles
  - or can guess Bob's random choice

# Asymmetric Encryption



- Requirements
  - public key for encryption  $E$
  - private key for decryption  $D$
  - $D(E(\textit{text})) = \textit{text}$
  - it is difficult to deduce  $D$  knowing  $E$ 
    - including high resistance to the “chosen plaintext” attack
  - sufficiently easy to generate  $E, D$  pair
- Everybody can encrypt message using  $E$
- Only Alice can decrypt it using  $D$ 
  - Bob sends encrypted message
  - Eve cannot decrypt it

## ~/ .ssh/known\_hosts

---

- Also /etc/ssh/ssh\_known\_hosts
- List of all known remote hosts
  - including their public keys
- Upon first connection, `ssh` cannot, by itself, verify the identity of the host
  - it should be done by the user
- Afterwards, `ssh` will be able to verify it is talking to the same remote system
  - and warn the user if something is wrong



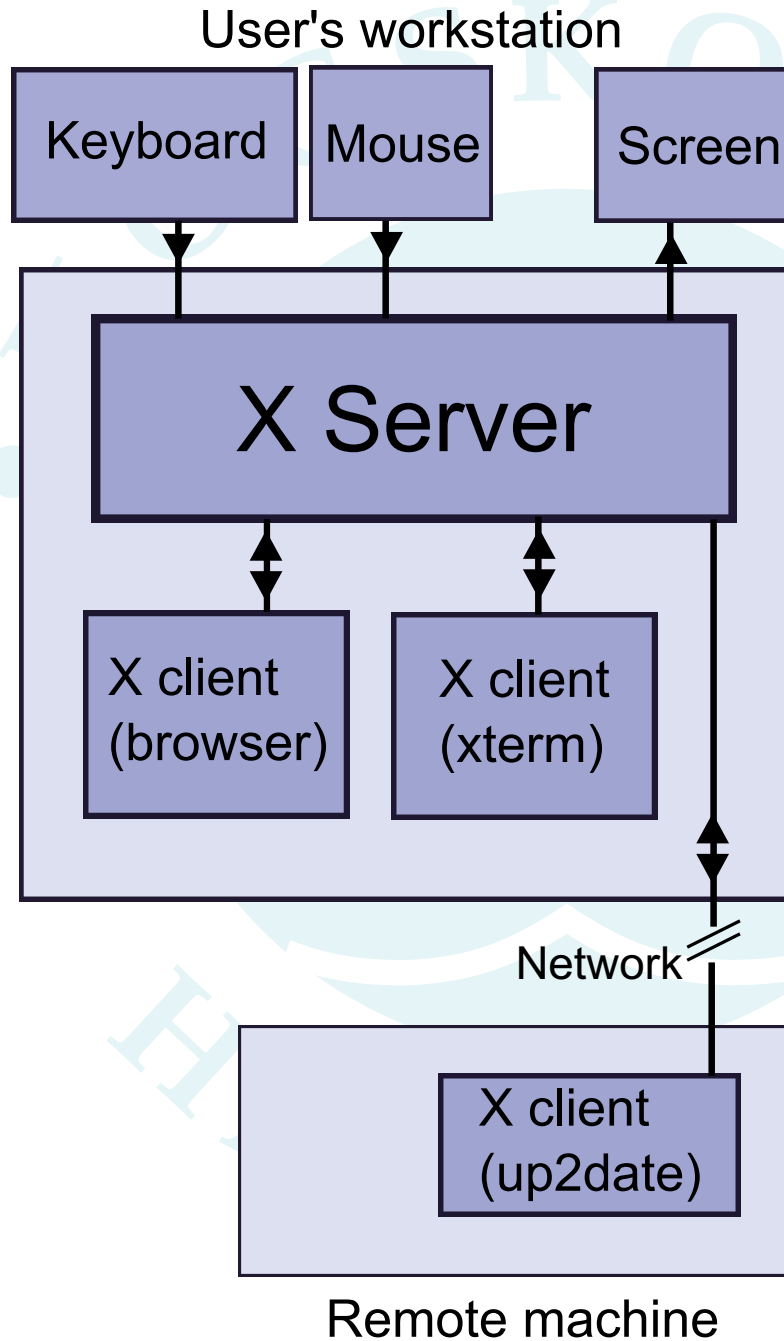
# Public Key Authentication

---



- Do not input password every time
  - authorise using asymmetric cryptography
- Generate a personal pair of keys
  - public & private
- Put public key on the server
  - this one does not need to be protected
- Keep private key safe
  - protected with a good passphrase
- Tell server that *everybody who knows this private key is considered trusted*
  - that's still better than password
- `ssh -i & ssh-agent`

# X11 Window System



# X11 Forwarding

---



- GUI environment used by GNU/Linux systems
  - architecture independent & network aware
  - using client-server model
- X11 forwarding allows remote system to display GUI objects on user's local machine
  - encrypting all traffic
  - *X server* needs to be running on local machine
- `sshd` acts as *X server* on remote host
  - GUI applications connect to it
  - traffic is tunnelled over encrypted connection
- `ssh` acts as GUI client
  - displaying GUI objects of remote applications

# Port Forwarding

---



- Similar mechanism can be used for forwarding arbitrary traffic
- Most commonly used for securing otherwise unencrypted protocols
  - POP3, SMTP, IRC, ...
- Local port forwarding
  - connections to local machine are transparently forwarded to the remote system
- Remote port forwarding
  - connections to remote machine are transparently forwarded to the local system

# Configuration

---



Intelligent  
Systems  
Lab

- AllowUsers & DenyUsers
  - AllowGroups & DenyGroups
- PasswordAuthentication
- PermitEmptyPasswords
- PermitRootLogin
- AllowAgentForwarding
- AllowTcpForwarding
- ListenAddress & Port
- Protocol
- ChrootDirectory
- Match User



Intelligent  
Systems  
Lab

**Questions?**

