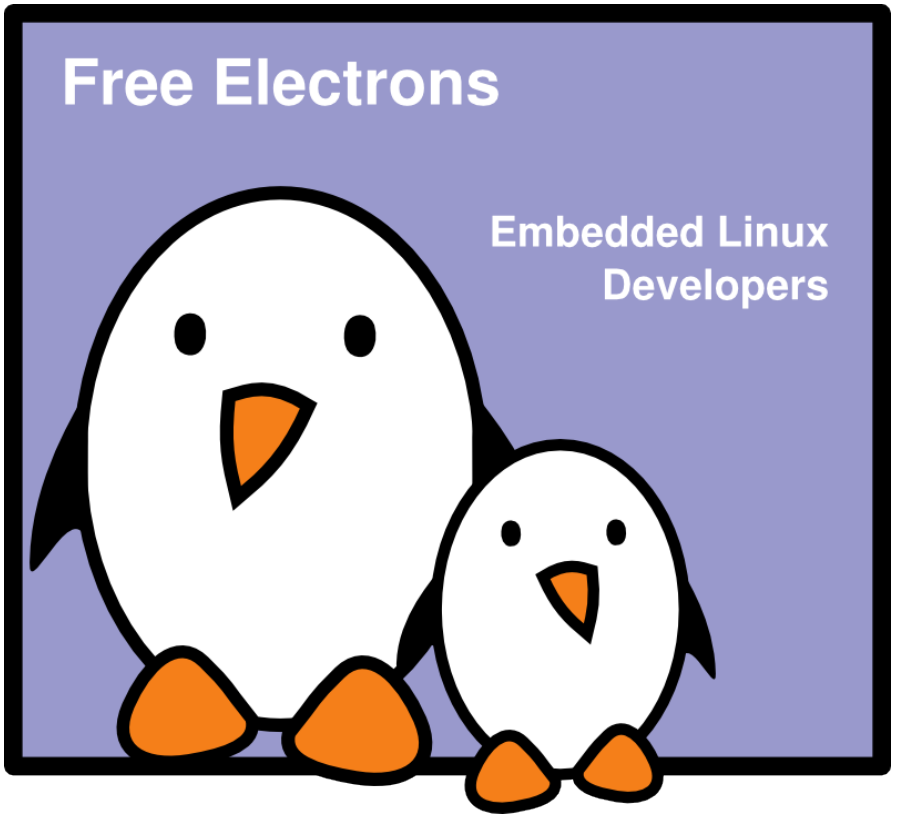
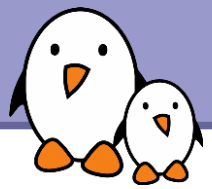
SSH

Thomas Petazzoni
Free Electrons



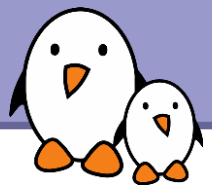


Rights to copy

Document sources, updates and translations:
http://free-electrons.com/docs/ssh

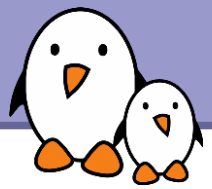
Corrections, suggestions, contributions and translations are welcome!

Latest update: Jan 29, 2009



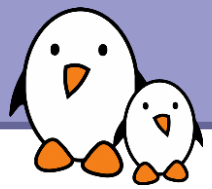
Introduction

- SSH stands for *Secure SHell*
- SSH is a secure communication protocol that allows remote login, file transfer and port tunneling, normalized by RFC 4251, 4252, 4253 and 4254.
- Replacement for telnet, rlogin, rsh, etc.
- On Linux, the main implementation is **OpenSSH**, with both the server and client programs
- A smaller implementation for embedded systems called **Dropbear** is also available
- On Windows, **Putty** is one of the free SSH client available.



Installation and basic usage

- OpenSSH is available as a package in all GNU/Linux distributions
- On Ubuntu, two packages are available
  - openssh-client, the client programs
  - openssh-server, the server program
- Connecting to an SSH server is as simple as `ssh username@hostname`
- ssh will prompt for the user password and log in to the remote system.

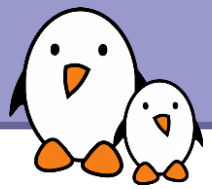


File transfer and X forwarding

- Files can be transferred using the scp client program

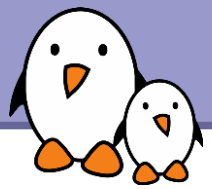
```
scp myfile1 myfile2 \
username@hostname:~/dest/directory/
scp -r mydirectory user@host:~/dest/
```

- With ssh `-X` option, one can tell ssh to enable X11 forwarding
  - It allows graphical applications run on the remote host to be displayed on the local screen
  - On the server, `X11Forwarding` must be enabled in the configuration file `/etc/ssh/sshd_config`.



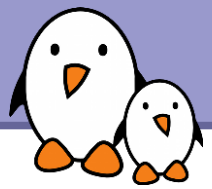
Remote execution

- ssh not only allows to connect to a remote host, but also allows remote execution of commands
  - `ssh user@host ls`
  - This is very useful in shell scripts, for example
- ssh is also used by other programs as a transport layer
  - rsync, the synchronisation tool, can work over ssh
    `rsync -e ssh ~/work user@workhost:~/work`
  - CVS, Subversion and most of the version control tools can work over SSH



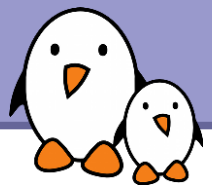
Skipping the password with keys

- An interesting feature of SSH is that you can bypass the password step by using cryptographic keys
- First, generate a private and public SSH key using `ssh-keygen -t dsa`
- It will prompt you for a passphrase, which will be required to «unlock» your private key everytime you use time
- The key has been generated in
  - `~/.ssh/id_dsa`, the private key, that no one should have access to
  - `~/.ssh/id_dsa.pub`, the public key, that you can transfer publicly to everybody



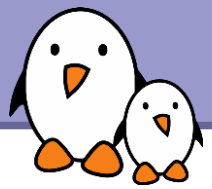
Skipping the password with keys (2)

- Now, you need to transfer the public key to the hosts you want to connect to
  - `ssh-copy-id user@host`
- The public key has been transferred to the remote host, and you should see it in `~/.ssh/authorized_keys` on the remote host
- Trying to login to the remote host should ask you the passphrase of the private key
- This allows to replace our dozens of different passwords by a single passphrase, which is easier to remember.



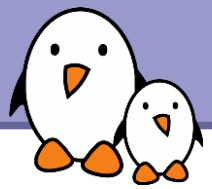
Skipping the password with keys (3)

- `ssh-agent` allows to avoid giving the passphrase at every login. It keeps the passphrase in memory, either forever or for a limited time
- Run the agent: `$(eval ssh-agent)`
  - Will run the `ssh-agent` program
  - Will set a few environment variables so that the other ssh programs can connect to the agent
- Give the passphrase to the agent: `ssh-add`
- The other ssh programs can now login to remote hosts that know about your public key without entering the password



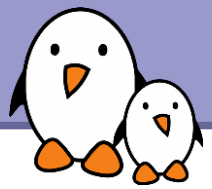
Skipping the password with keys (4)

- The environment variables set by ssh-agent disappear when you exit the current shell
- The best solution is to start the ssh-agent before starting the X server so that all your applications will have access to these environment variables
- This is usually done by default on most distributions, including Ubuntu
  - The file `/etc/X11/Xsession.options` sets the `use-ssh-agent` option
  - A script in `/etc/X11/Xsession.d/` starts the agent if the `use-ssh-agent` option is set



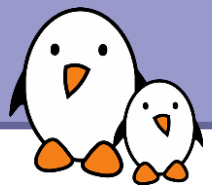
Skipping the password with keys (5)

- The process of telling the agent your passphrase can be further improved by
  - Installing a graphical ssh-add program: ssh-askpass-gnome for Gnome or ksshaskpass for KDE (only available in the next Ubuntu version)
  - Running ssh-add automatically when the graphical environment starts. The exact configuration depends on your window manager.



Port tunneling

- SSH can also be used to tunnel ports
- Create a local port that connects to a remote host through a SSH connection to another host
  - `ssh -L 12345:localhost:25 user@host`
  - Any connection on the local port 12345 will in fact reach port 25 on the destination, through an encrypted tunnel
- Create a remote port that connects to a host through a SSH connection to localhost
  - `ssh -R 4242:kernel.org:80 user@host`
  - Any connection on the remote host port 4242 will in fact reach port 80 of kernel.org through an encrypted tunnel



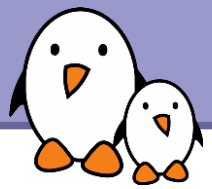
Configuration file

- SSH stores a configuration file in `~/.ssh/config`
- It can be used to set global options, but also per-host options, like
  - `Host openmoko`
  - `HostName 192.168.0.202`
  - `User root`
- Using these options, running "`ssh openmoko`" will connect automatically to IP 192.168.0.202 with the root login.

Practical lab – Using SSH

Time to start `Lab` !

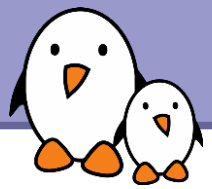
- Ask your neighbor to create an account for you
- Login to your neighbor system using ssh
- Set up the keys to login without entering any password



Thanks

To people who sent corrections, suggestions or improvements

- Guillaume Lelarge



Related documents

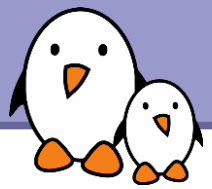
All the technical presentations and training materials created and used by Free Electrons, available under a free documentation license (more than 1500 pages!).

http://free-electrons.com/training

- Introduction to Unix and GNU/Linux
- Embedded Linux kernel and driver development
- Free Software tools for embedded Linux systems
- Audio in embedded Linux systems
- Multimedia in embedded Linux systems

http://free-electrons.com/articles

- Advantages of Free Software in embedded systems
- Embedded Linux optimizations
- Embedded Linux from Scratch... in 40 min!
- Linux USB drivers
- Real-time in embedded Linux systems
- Introduction to uClinux
- Linux on TI OMAP processors
- Free Software development tools
- Java in embedded Linux systems
- Introduction to GNU/Linux and Free Software
- Linux and ecology
- What's new in Linux 2.6?
- How to port Linux on a new PDA



How to help

If you support this work, you can help ...

- By sending corrections, suggestions, contributions and translations
- By asking your organization to order training sessions performed by the author of these documents (see http://free-electrons.com/training)
- By speaking about it to your friends, colleagues and local Free Software community.
- By adding links to our on-line materials on your website, to increase their visibility in search engine results.

Free Electrons services

Embedded Linux Training

Unix and GNU/Linux basics
Linux kernel and drivers development
Real-time Linux
uClinux
Development and profiling tools
Lightweight tools for embedded systems
Root filesystem creation
Audio and multimedia
System optimization

Custom Development

System integration
Embedded Linux demos and prototypes
System optimization
Linux kernel drivers
Application and interface development

Consulting

Help in decision making
System architecture
Identification of suitable technologies
Managing licensing requirements
System design and performance review

Technical Support

Development tool and application support
Issue investigation and solution follow-up with mainstream developers
Help getting started

http://free-electrons.com

Free Electrons
Free Software for Embedded Systems