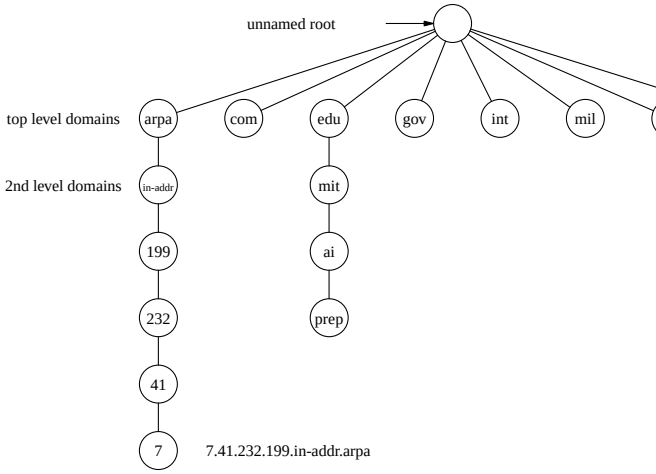
DNS

- is a distributed database that is used by TCP/IP applications to map between hostnames and IP addresses
- no single site knows all the information
- DNS name space is hierarchical

DNS



DNS

- diagram shows how the IP address `199.232.41.7` is mapped onto `prep.ai.mit.edu`
- note that DNS dictates that a FQDN must not exceed 255 octets and a machine name must not exceed 63 octets
- note that the DNS map is broken into zones
  - com, org, mil
  - requests for names into IP addresses are resolved by hierarchy

DNS

- note the clever technique of reverse name lookup
  - `199.232.41.7`
- use the same algorithm for name lookup but with numbers
  - look up the domain `7`, then `41`, then `232` and `199`

Internet Cafés

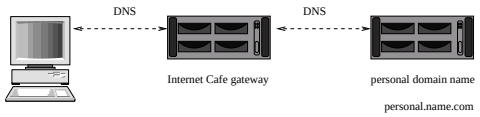
- may consist of wireless networks which uses a gateway
  - which may rewrite HTML pages to include a house style frame
- which may result in including advertising around the edge of your browser screen

Tunnelling IP over DNS

- IP can be run over any piece of digital string
- Internet café users became increasingly annoyed with the force fed advertising
- devised a mechanism whereby this could be by-passed

Architecture of IP over DNS

-



Architecture of IP over DNS

- protocol stack now looks like:

HTTP
TCP and UDP
IP
DNS
TCP and UDP
IP
802.11g

Architecture of IP over DNS

- notice that the Internet café no longer sees any of the computers
  - IP, TCP, UDP or HTTP traffic
  - as it is all encapsulated within the DNS packets
- users machine has to run a DNS over IP protocol stack
- which takes in IP packets and issues requests which take the form:
  - `xyz123.personal.name.com`
  - *any 63 ASCII characters in range* `a` to `z` and `0` to `9` added to `.personal.name.com`

Tunnelling IP over DNS

- Internet café DNS sees requests made on weird hostnames at `.personal.name.com` and forwards these values to users remote machine at `.personal.name.com`
  - which also runs the same DNS over IP protocol and which translates these weird 63 byte names into an IP datagram
    - and transmits the datagram on behalf of the Internet Café user
    - it returns the result
- Internet café gets the DNS reply and returns the reply to the user

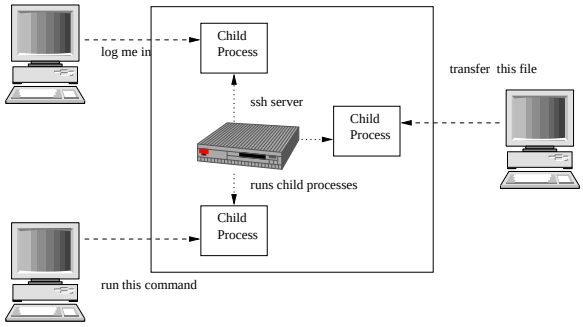
Tunnelling IP over DNS

- ultimately the DNS over IP will be much slower than using normal TCP/IP over 802.11g
  - but it does work and is an ingenious idea
- it has serious implications in security of networks as it renders inexpensive firewalls useless (at the time of writing)
  - why?

SSH

- Secure SHell is a popular software approach to network security
  - operates at the application layer
- offers transparent encryption, authentication and integrity of data
- provides command line commands:
  - `ssh`, `sftp`, `scp` and `slogin`
- there are also a number of influential tools which use the `ssh` protocol
  - `rsync`, `git` and `sshfs`

SSH architecture



SSH Protocol

- provides Authentication
  - reliably determines someones identity using public/private key
- provides encryption
  - scrambles data as it passes across the network
- provides integrity
  - guarentees data travels across the network unaltered

SSH Port Forwarding

- the ssh tools: `ssh`, `slogin`, `scp` and `sftp` are useful enough to warrent investment in time with ssh
- however the port forwarding capability ensures that ssh enters non command line networking!
- any port can be forwarded across ssh

SSH Port Forwarding

- certain protocols transmit usernames/passwords in plaintext or using weak passwords
  - imap, pop3 and vnc, X windows
- ssh can be used to harden these very useful protocols

SSH examples

- connecting to a remote machine

```
ssh mcgreg.comp.glam.ac.uk
Password:

Linux mcgreg i686 GNU/Linux
Last login: Tue Feb  8 10:47:44

fred@mcgreg:~/$ exit
```


SSH examples

- using a command line ftp equivalent

```
sftp mcgreg.comp.glam.ac.uk
Password:
sftp> dir
sftp> get foo.ps
sftp> quit
```


X Windows Port forwarding

- GNU/Linux allows graphical applications to be run remotely
  - remote desktop per application
    - as well as per desktop (using vnc)

```
ssh -X mcgreg.comp.glam.ac.uk
Password:

Linux mcgreg i686 GNU/Linux
Last login: Tue Feb  8 10:47:44

fred@mcgreg:~/$ xterm
fred@mcgreg:~/$ exit
```

- ssh is forwarding all X traffic across port 22

Browsing the web through an ssh connection

- suppose we want read the web pages of `floppsie.comp.glam.ac.uk` securely

```
ssh -g -A -X -N -T -L2001:localhost:80 floppsie.comp.glam
```

- which means create a secure link between port 2001 on localhost and port 80 on floppsie.comp.glam.ac.uk

```
telnet localhost 2001
get index.html
```

- or `http://localhost:2001/index.html`

Flag meanings

- `-g` allow connections to forwarding `ssh` sessions
- `-A` enable forwarding of authentication
- `-X` forward X windows traffic
- `-N` do not execute a remote command
- `-T` disable tty allocation (so a command can be run in background or by another application)

SSH through an untrusted proxy ssh server

- on your local machine you type:
- `ssh -g -A -X -N -T -L2001:trusted.com:22 untrusted.proxy.`
- which says open a secure connection starting at port 2001 on our local machine
  - which provides a connection between untrusted.proxy.com and trusted.com on port 22
  - the flags turn all port forwarding capability

SSH through an untrusted proxy ssh server

- and in another terminal window type:
- `ssh -v -g -A -X -p 2001 localhost`
- which now opens up a connection between your keyboard and localhost:2001
  - effectively giving you a secure encrypted connection to trusted.com:22

Laboratory work

- try out all the examples presented in todays lecture (except the floppsie example)

Python and SSH

- type in the following code and analyse what it does!

Python and SSH

```
#!/usr/bin/python

#  secure shell pipe module

import os
import sys
from socket import *

localPortNo=8000
maxTries=10

#  createTCPSocketSSH - creates a secure TCP socket betwe
#                       localhost:localPort and
#                       remoteHostname:remotePort

def createTCPSocketSSH (remoteHostname, remotePort=22,
                        localPort=-1):
```


Python and SSH

```
    global localPortNo
    if localPort == -1:
        localPort = localPortNo
        localPortNo = localPortNo+1
    tryNo = 1
    while 1:
        command = "ssh -f -g -A -X -N -T -L%d:localhost:%
                  (localPort, remotePort, remoteHostname)
        print command
        result = os.system(command)
        if result == 0:
            break
        localPortNo = localPortNo+1
        tryNo = tryNo + 1
        if tryNo == maxTries:
            os.exit(1)
```


Python and SSH

```
    # create a TCP socket which connects to our ssh pipe
    s = socket(AF_INET, SOCK_STREAM)
    s.connect(("localhost", localPort))
    return s


s = createTCPSocketSSH("mcgreg.comp.glam.ac.uk", 80)
s.send('get index.html\n')
print s.recv(1024)
```

Tutorial

- firstly reaquaint yourself with ssh by logging into `mcgreg.comp.glam.ac.uk`
  - then exit from this server
- now `sftp` a file from your desktop to `mcgreg.comp.glam.ac.uk` to your `$HOME` directory
- now `ssh` to `mcgreg.comp.glam.ac.uk` with X forwarding enabled
  - and run `firefox` on `mcgreg.comp.glam.ac.uk` to confirm that this works
- now create a secure link between `localhost:8080` and `mcgreg.comp.glam.ac.uk:80`
  - you will need to use the following options on the `ssh` command line `-g -A -X -N -T` as well as the port and hostname parameters
- do not kill the previous `ssh` command and now, using a browser on your local machine open up a URL which matches `http://localhost:8080`
  - can you read the web page on mcgreg?
  - draw a diagram clearly showing the various hostname:port network connections and clearly highlight which components are encrypted

mrun and the CSN

- open up a terminal and type
- ```
  $ mrun -h
  $ mrun -L
  ```
- `mrun` is a program for distributing a parallel application onto multiple processors
  - it uses `ssh` to manage connections between selected processors
- now download and build some examples
- ```
  $ cd Desktop
  $ tar zxf /usr/local/share/csn/examples.tar.gz
  $ cd examples
  $ make
  ```
- now examine all the *filename*`.par` files using an editor or filemanager of your choice

mrun and the CSN

- examine the `hostname.par` first, what is it doing?
- try running it using `mrun`, hint use the command line and type:
- ```
  $ mrun -f hostname.par
  ```
- now try the same for `echo.par` examine the file and observe its behaviour
  ```

`mrun` and the CSN

- try running `hello.par`, you may need to alter the architecture field which is currently set as `(localhost)` to `(j203)`
- if you are taking operating systems examine the C files `txhello.c`, `rxhello.c`
- if there is still time available examine `manager.c` and `worker.c`
- further reading ⟨`http://floppsie.comp.glam.ac.uk/csn/csn.html`⟩

Tutorial (Exam question 2010 3b)

- (b) In the eager reader scenario within a device driver, how might the device driver determine the length of 3 distinct buffers to contain an arriving IEEE 802.3 packet containing an IP, TCP and application layer packet?

(10 marks)

Tutorial

- (i) how many bytes can a single outgoing DNS request carry as part of the hostname component of the FQDN?
- (ii) what would happen if identical FQDN requests were made to the remote DNS server?
- (iii) how would you prevent this from occuring?

Tutorial: Question 2: 2010

- "Tunnelling IP over DNS will be one of the biggest security
 problems of the future", discuss.

(20 marks)

Tutorial: Question 8 2007

- (a) "Network protocols are intrinsically related to operating systems". Critically appraise this statement in relation to the LTSP project.

  (12 marks)

- (b) Comment on the suitability of LTSP in an office and University laboratory environment.

  (8 marks)

Tutorial: Compare NFS to RTSP

- what makes RTSP useful for streaming video content?
  - how does this compare to NFS?
- what are the pros and cons of various syslinux options which might be used to on a laboratory computer?

Tutorial: Plan 9 and LTSP

- What are the elements of design were utilised in Plan 9 and found in LTSP?
- What were their likely motivation?