Linux Networking:
- network services

David Morgan

Client and server: matched pairs

Client process  \[\text{inter-process communication}\]  Server process

© David Morgan 2003-2009
OK as long as there's a way to talk

Client process

Server process

OK as long as there's a way to talk

Client process

Server process
Trans-net way to talk: socket programming

- a communications interface/mechanism
- like IPC (inter-process communications)
- but generalized to span machines (inter-machine-inter-process)
- coded like file handles
- sockets correspond to service “ports” of TCP and UDP

Ports and conversations

<table>
<thead>
<tr>
<th>Source Address</th>
<th>Destination Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td>Destination Port</td>
</tr>
</tbody>
</table>

TCP’s Data Payload

Uniquely identifies a process-to-process conversation

© David Morgan 2003-2009
Distinction: machine vs process

Client process

Server process

client machine??

server machine??

Client processes

Server processes
What are port numbers?

ports' numbers?

What are port numbers?

or processes' numbers?
Service processes: 2 ways to run

- As servers in their own right
  - contain socket API code themselves
  - actively listen for incoming connections on a port
- Under control of xinetd “super server”
  - don’t contain socket code
  - don’t listen for connections
  - xinetd listens on their behalf

Servers in their own right
Managed by xinetd

- client program
- socket API
- transport
- network
- data link
- physical

- xinetd program
- socket API
- transport
- network
- data link
- physical

Examples, run on their own

- httpd  web/browsing service
- smtp   message transfer (mail) service
Examples, managed by xinetd

- ftp file transfer service
- telnet remote login service
- pop3 message retrieval (mail) service

xinetd operation

- a server that manages servers
- listens to other servers’ ports, for them
- responds to an incoming connection
  - starts server associated with connection’s port
  - connects server’s standard I/O to that port
  - lets server die when connection closes
xinetd purposes

- economize number of services running
- provide access control to managed services
- provide logging to managed services
- let any program be a special-purpose server

xinetd configuration

- `/etc/xinetd.conf`
- defines services managed by xinetd

```plaintext
service <service name>
{
    attribute = value
    .
    .
    .
}
```
Sample configuration: telnet

```plaintext
service telnet {
    disable = no
    flags   = REUSE
    socket_type = stream
    wait    = no
    user    = root
    server  = /usr/sbin/in.telnetd
    log_on_failure += USERID
}
```

Default RedHat xinetd.conf

```plaintext
# Simple configuration file for xinetd
# Some defaults, and include /etc/xinetd.d/
defaults {
    instances = 60
    log_type  = SYSLOG authpriv
    log_on_success = HOST PID
    log_on_failure = HOST
    cps        = 25 30
}
```

`includedir /etc/xinetd.d`
includedir=/etc/xinetd.d attribute

- all files in /etc/xinetd.d included by extension
- you can specify services one per file
- packages drop config files in /etc/xinetd.d to be picked up automatically by xinetd

Default directories and files

```
/   \
  /etc
    /httpd.conf
    /xinetd.d
      telnet  configures telnet
      wu_ftpd  config ftp
      ipop3   config pop mail
      rsh     config remote shell
      many others
```
Required attributes
(others optional)

```
service <service name>
{
    socket_type = …
    user = …
    server = …
    wait = …
}
```

Required attributes

- **socket_type**  stream/dgram - TCP or UDP
- **user**        username - user as whom service should run
- **server**      path – path to server executable
- **wait**        yes/no – should xinetd butt out till server quits?
Other attributes

- **disable**
  - yes/no – disables service
- **log_on_failure**
  - Special – info to log for access denials
- **no_access**
  - matchlist – deny to specified addresses
- **only_from**
  - matchlist – from specified addresses only
- **bind**
  - interface – where to make service available
- **log_type**
  - special – choose log file or syslog
- **redirect**
  - ipaddr – pass traffic to another host

Turning self-managed services on and off

- Services re-read configuration files when restarted
- Restarting
  - `/etc/rc.d/init.d/<script for service> restart` or
  - `service <script for service> restart`
Turning xinetd-managed services on and off

- Adjust service’s config stanza (in xinetd.conf) or file (in /etc/xinetd.d)
  - disable = off (to turn on)
  - disable = on (to turn off)

- Restart xinetd
  - /etc/rc.d/init.d/xinetd start or
  - service xinetd start

The services file

- /etc/services
- maps service names to port numbers
- used by several standard library routines
- e.g., “telnet mybox smtp” instead of “telnet somebox 25”
# /etc/services:

- **ftp-data**: 20/tcp
- **ftp**: 21/tcp
- **telnet**: 23/tcp
- **smtp**: 25/tcp, mail
- **domain**: 53/tcp, nameserver (name-domain server)
- **domain**: 53/udp, nameserver

## Detecting services & conversations

- `ps ax | grep <service/daemon name>`
  - snapshot of active processes/services
- `netstat -ap`
  - displays services corresponding to ports
Talking to a service: SMTP

- Uses port 25
- telnet <mailserver IP> 25
- Try SMTP commands
  - HELO
  - MAIL FROM:<sender>
  - RCPT TO:<recipient>
  - DATA
  - QUIT

Talking to a service: POP3

- Uses port 110
- telnet <mailserver IP> 110
- Try POP3 commands
  - USER username
  - PASS password
  - LIST
  - RETR msg#
  - DELE msg#
  - QUIT
Talking to other services

- FTP
  - telnet <server IP> 21
  - RFC 959
- SSH
  - telnet <server IP> 22

Network Services – quick start

- to serve web pages, out of the box*
  - Put them in /var/www/html
- to serve files, out of the box
  - Put them in /var/ftp/pub

* RedHat, having chosen “everything”
  package installation choice