Raspberry Pi as Black Box

Raspberry Pi “Black Box”
Download Test and Logging Tool

Ian E. Gorman

Ottawa Canada Linux Users Group
July 7, 2016

National Capital Freenet

- NCF, in Ottawa Ontario, is a local Internet Service Provider
- NCF is a Bell reseller for DSL service
- Bell provides NCF with test data and technical support to diagnose and eliminate problems that may originate in Bell equipment or services
- Bell support does not extend to problems originating from NCF or from NCF users

Problem

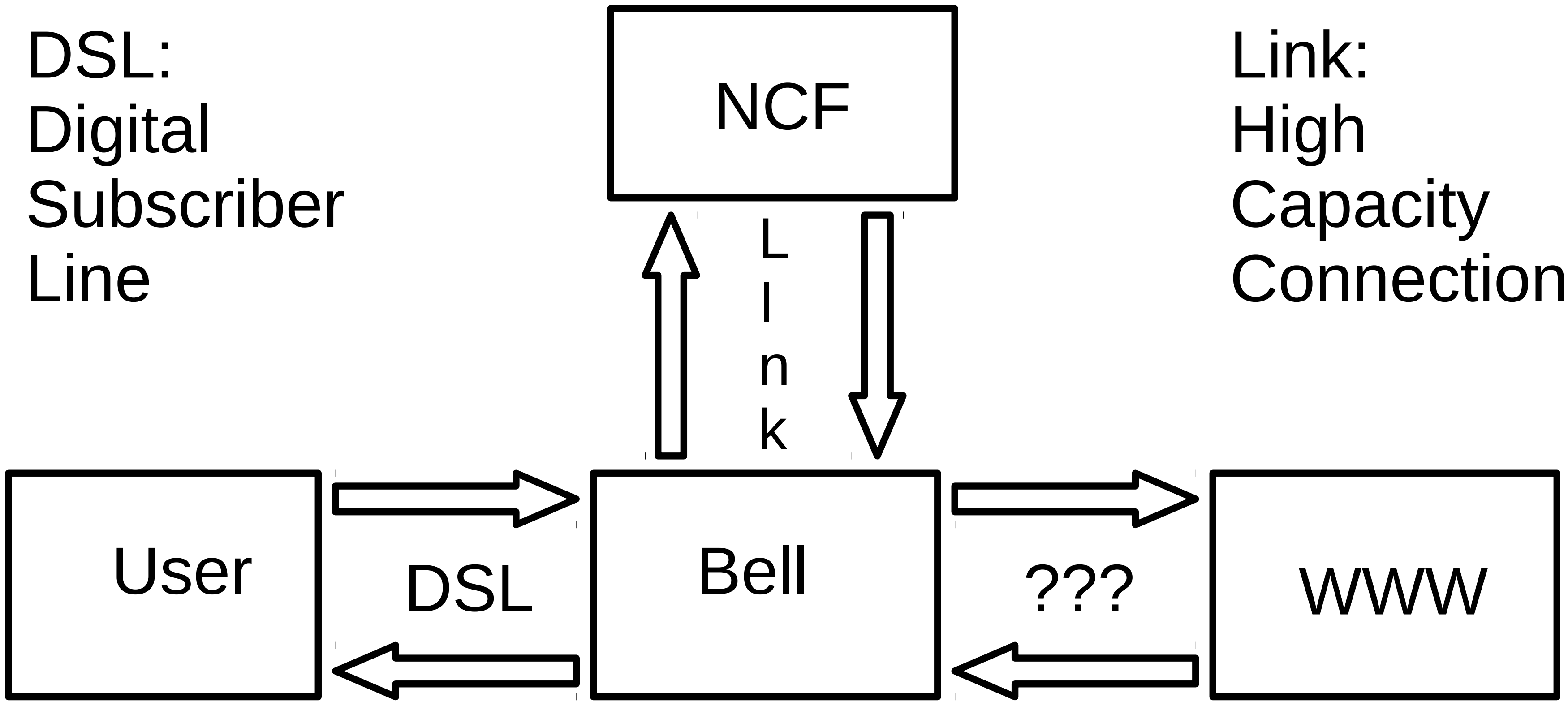
- Some NCF users report download speeds much below expectations
- Test data may indicate that, notwithstanding low speeds, a user has a good and properly functioning DSL connection
- Speeds experienced by the user are influenced by factors other than the quality of the DSL connection
- Many users do not have the expertise to assist NCF in testing

NCF Connection to WWW

Bell provides communications infrastructure, used by NCF to provide Internet services to end user.

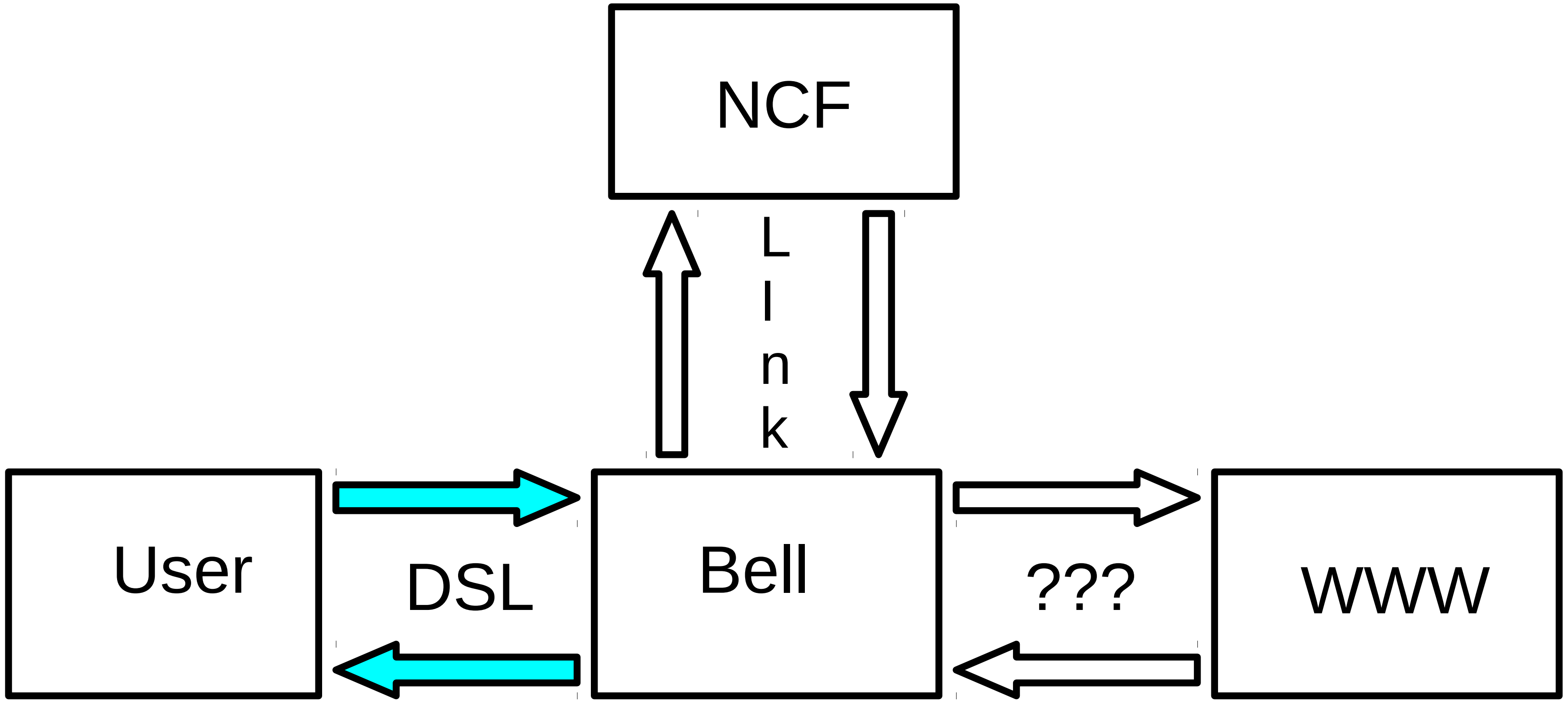
DSL:
Digital
Subscriber
Line

Link:
High
Capacity
Connection



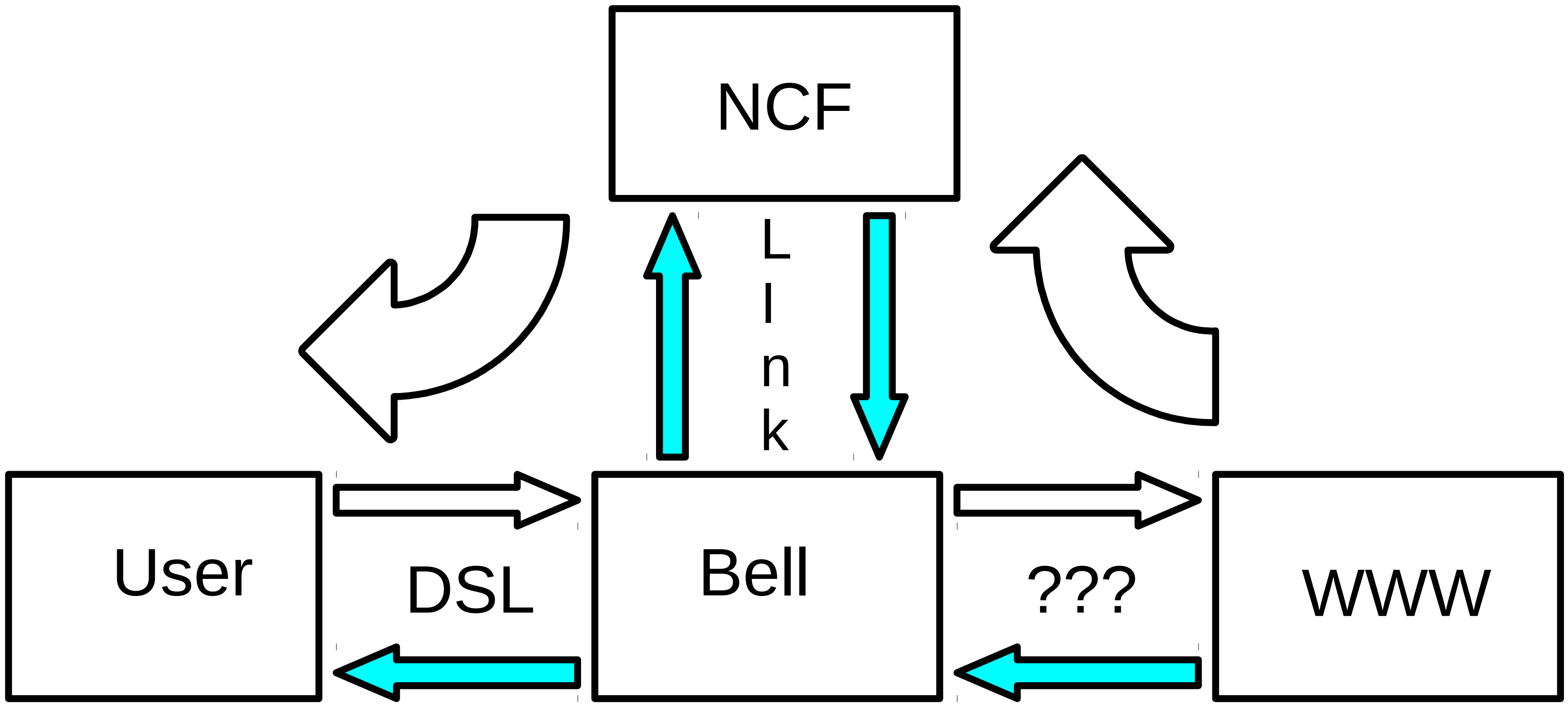
Scope of Bell DSL Test Data

Bell provides NCF with some very useful diagnostic data for the DSL part of the communication path.



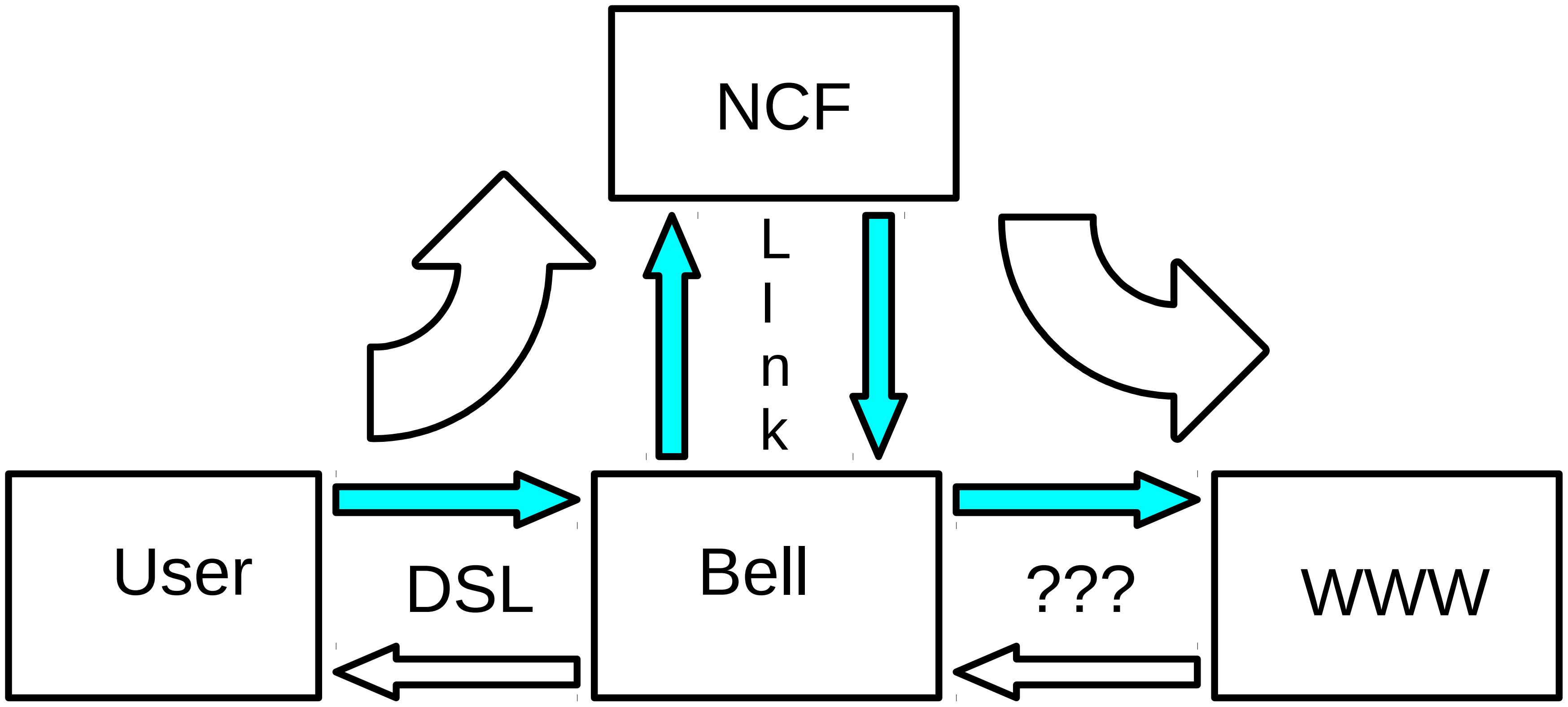
User Download from WWW

Download comes to Bell, goes to NCF (and NCF user account), back to Bell and then to the user.



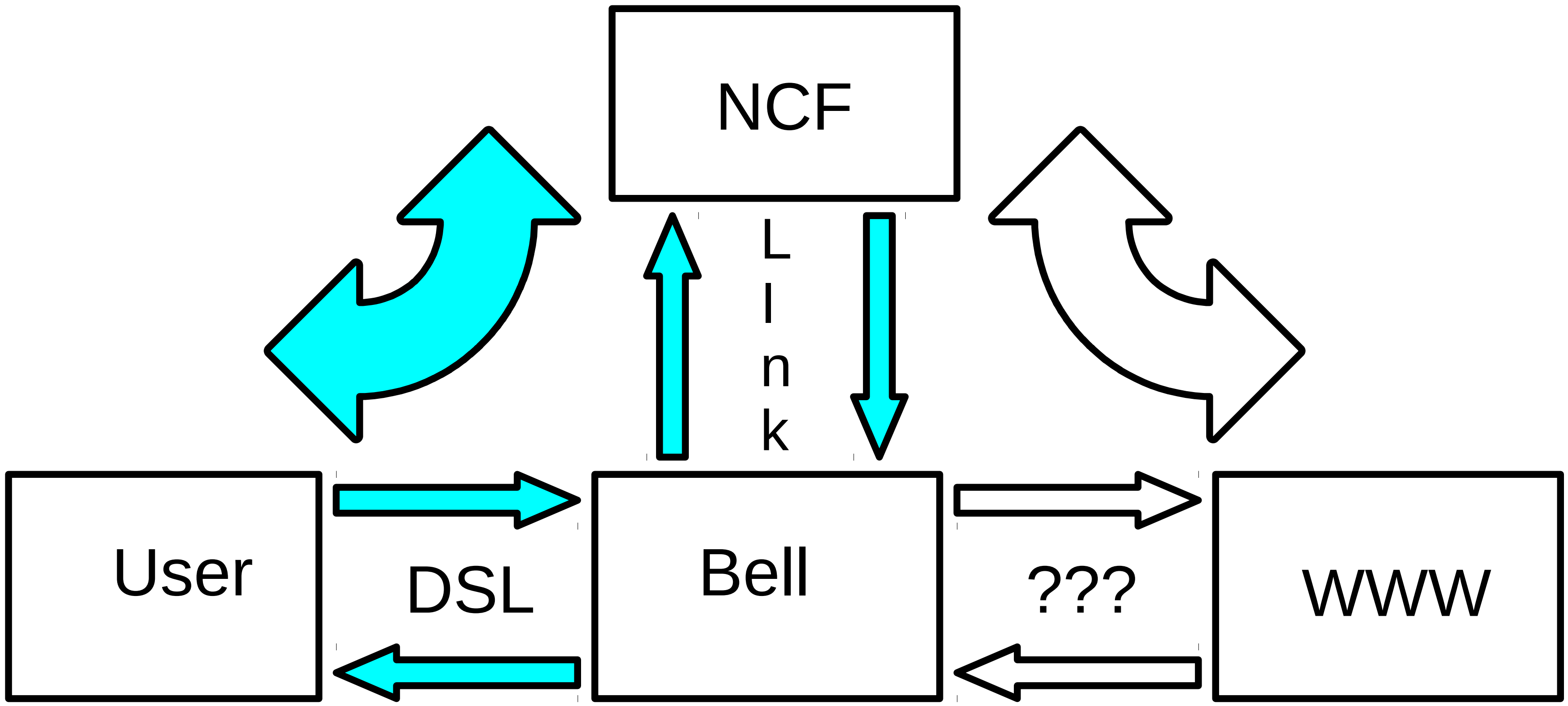
User Upload to WWW

Upload goes to Bell, then to NCF (and NCF user account), back to Bell and then to the WWW.



DNS and Protocol

Unidirectional transfers of large amounts of data typically require small transfers the other way.



A Black Box Solution

- Lend a black box to the user
  - User plugs it in
  - Black box runs tests, perhaps for several days
  - User unplugs and returns black box
- NCF analyses test data
  - Produces a report, with graphs, from the test log
  - Draws conclusions about extent, nature, and possible location of problem

Black Box Operation

- Test Kit
  - One Raspberry Pi (in a black case!)
    - Test software installed in the Raspberry Pi
  - One 5 volt power block for Raspberry Pi
  - One Ethernet cable
- Testing
  - User plugs black box into router (with Ethernet cable) and wall (with power block), watches the lights go on.
  - When the lights go out, user unplugs the box and sends everything back to NCF
  - NCF runs the Awk and R scripts to produce a report

Black Box Test Software

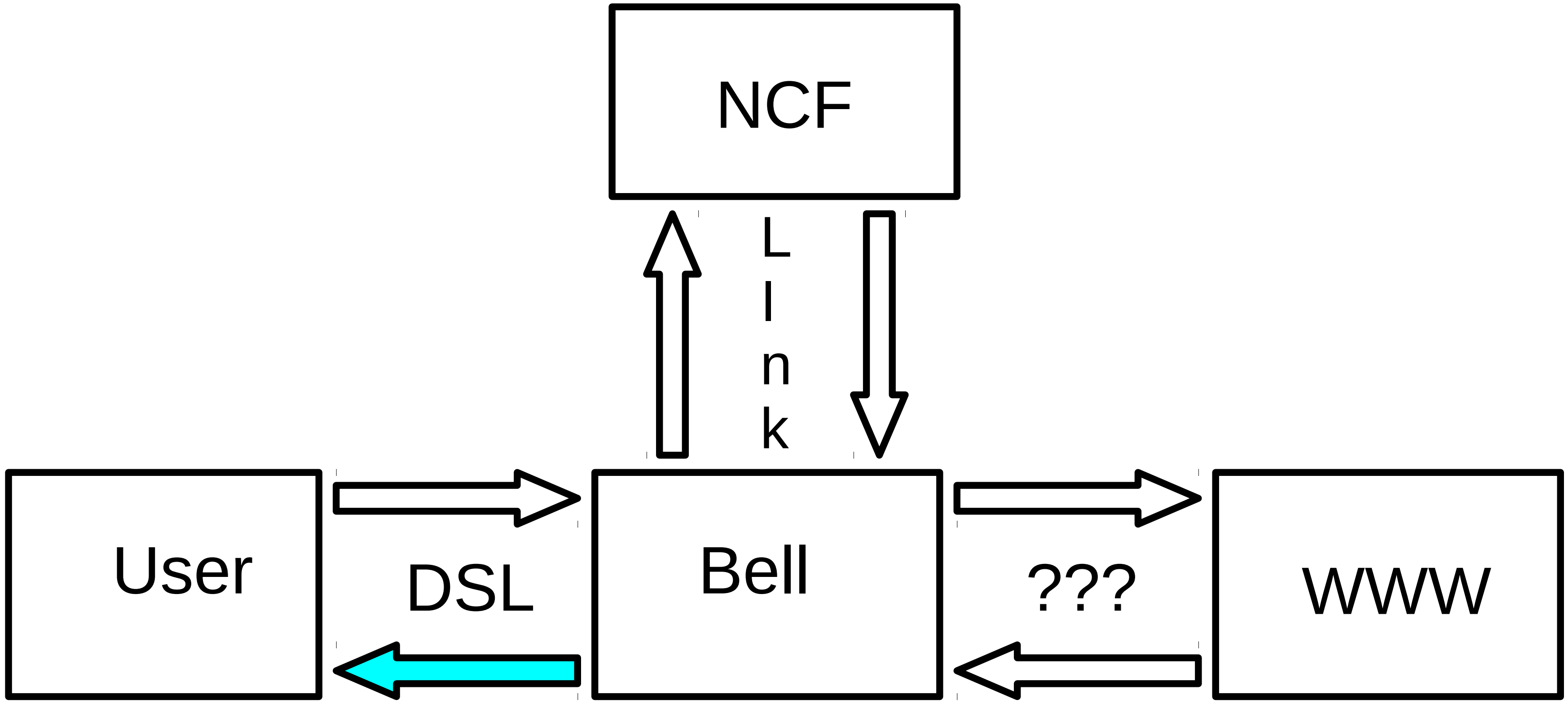
- Test runs
  - Shell script to run and log one test
  - Shell script to repeat the tests
  - Shell script to act as hook for crontab entry
  - Crontab entry
- Processing
  - Awk script to translate log to one line per download
  - R script to produce report
- Setup
  - Python script to generate test files for download

Progressive Test Strategy

- Link (from Bell data)
- DSL (from Bell data)
- User Equipment + DSL
  - User modem is tested, user takes black box home
  - Black box records downloads from Bell
- User Equipment + DSL + Link + NCF
  - Black box records downloads from NCF
- User Equipment + DSL + Link + NCF + WWW
  - Black box records downloads from WWW

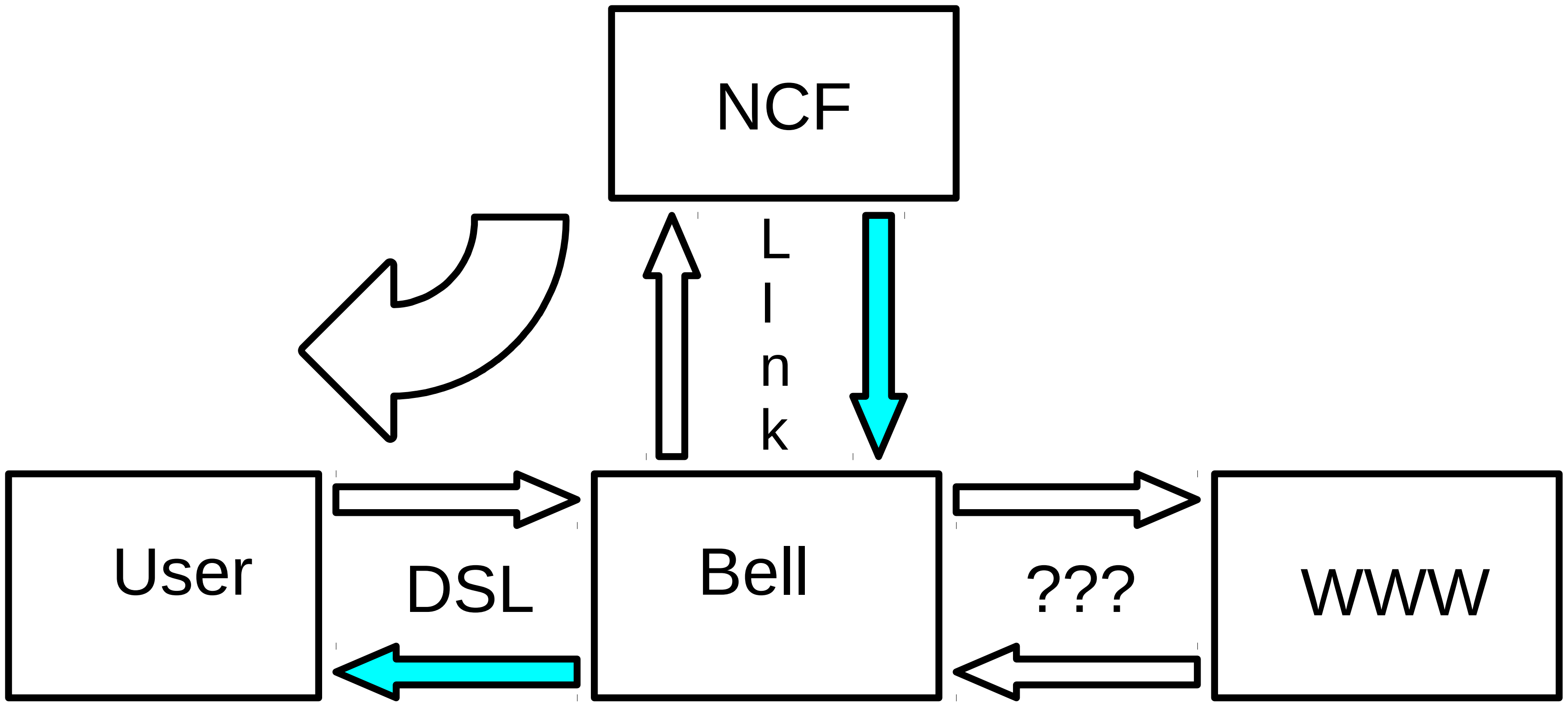
Download Test from Bell

User equipment and DSL path can be tested by downloading data from the Bell system



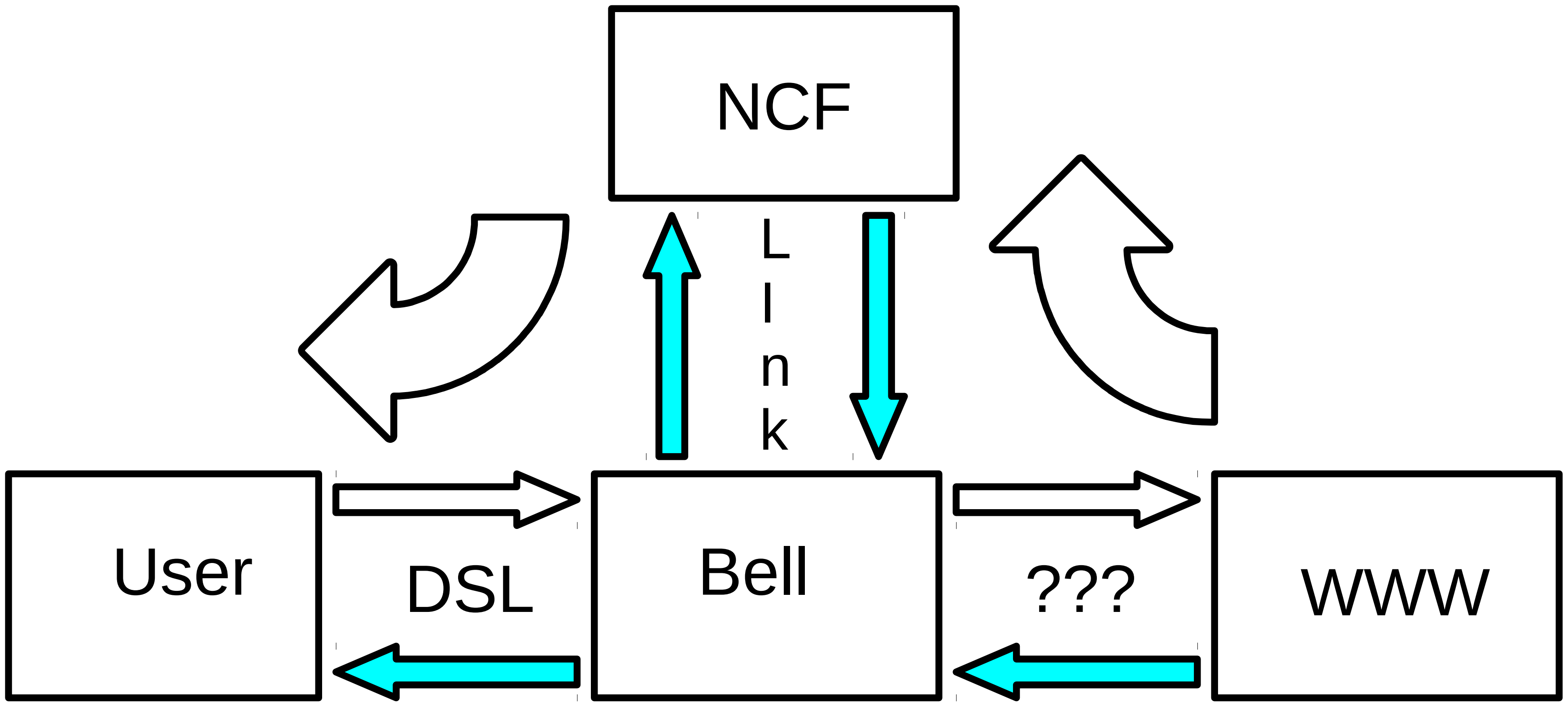
Download Test from NCF

User equipment, DSL path, and Link path can be tested by downloading data from the NCF system



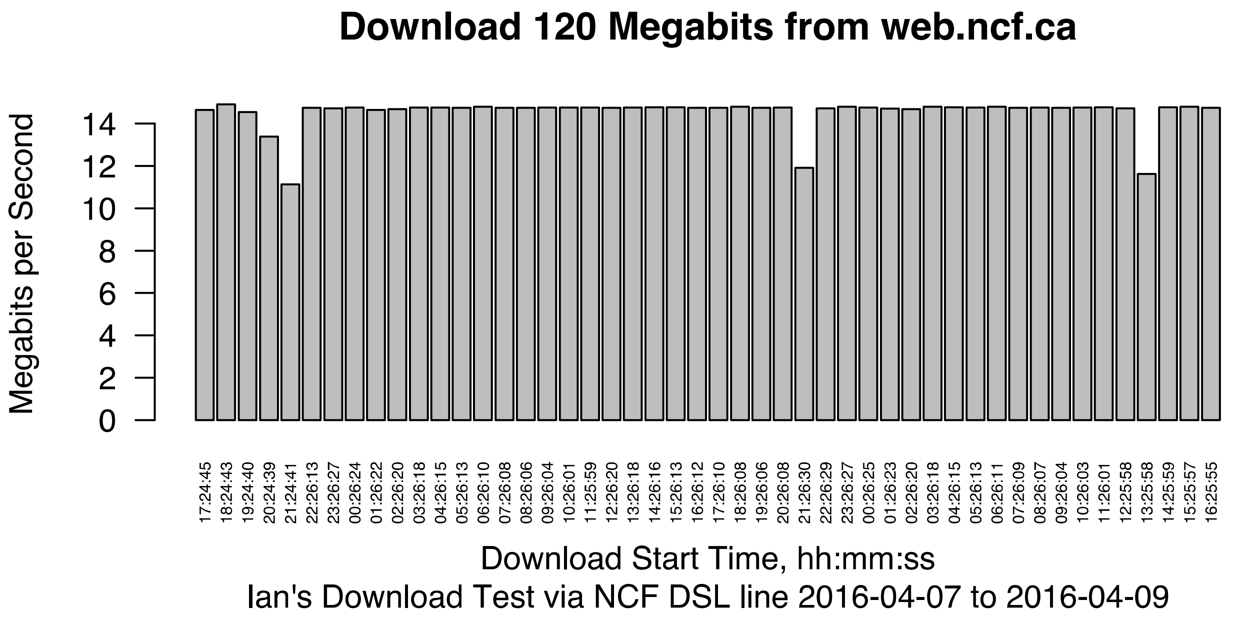
Download Test from WWW

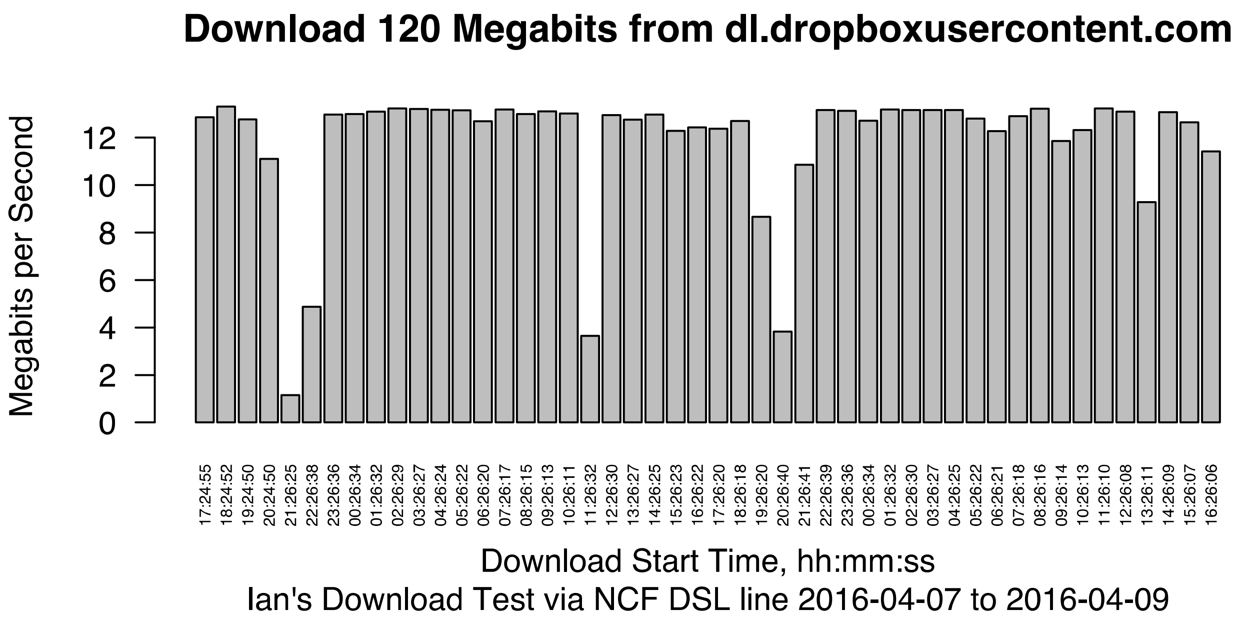
User equipment plus entire path can be tested by downloading data from the WWW



Bar Chart

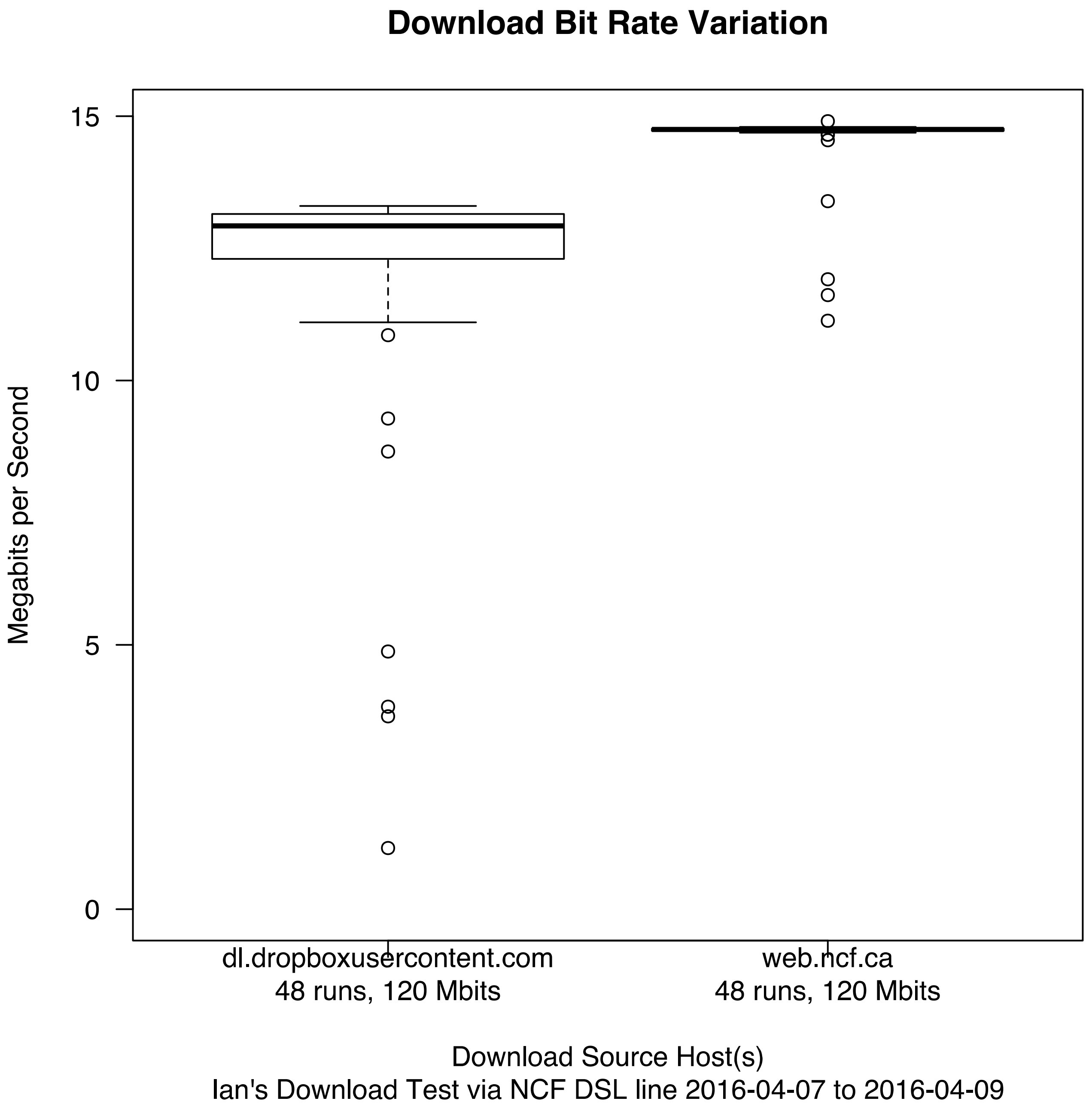
Compare speeds from up to three download sources





Scatter Chart

Compare variation from up to three download sources



Test Software in Raspberry Pi

- User plugs Raspberry Pi in to modem and power, without keyboard or mouse
- Crontab starts a control script in $HOME (~pi) and redirects output to a file in $HOME
- Control script takes no parameters but has internal parameters that can be set to run the desired tests
- Control script runs a test script as specified by the internal parameters, to append data to a log file
- Test script script exits to the control script
- Control script shuts the Raspberry Pi down

Test Analysis with Raspberry Pi

- Technician starts Raspberry Pi with keyboard and mouse, but no network connection
- Technician logs in before test failure shuts Raspberry Pi down, and stops current test run
- Awk program converts test log to report file with one line per download.
- R program produces graphs and a summary from the report file
- Raspberry Pi can be connected to network to transfer graphs and reports

crontab Entry

```
# crontab entry to run download speed test and then shut
down
# scripts require 'bash' and will fail in default Raspbian
shell
# see crontab(5) for default shell
SHELL=/bin/bash
#
# redirection must be done within the crontab command
line, see cron(8)
# see crontab(5) for comand line format
@reboot~/speedtest/speedtest.cron.sh >>
~/speedtest.log 2>&1
```

Control Script (excerpt 1)

```
# wait for system startup and DHCP to complete
sleep 120

# Run the test script with the parameters,
environment, and termination options
# created by this script
"${SCRIPTNAME%/*}"/speedtest.sh "$URL_1"
"$URL_2"  "$URL_3" "$INTERVAL" "$COUNT"
```

Control Script (excerpt 2)

Shutdown and error management:

```
# In normal operation, shutdown occurs 2 minutes after other script returns
# Shutdown can be avoided by deleting the file ${SHUTDOWNFLAGFILE}
#
SHUTDOWNFLAGFILE=~/".shutdown.after.speedtest"
:>"${SHUTDOWNFLAGFILE}"
echo "Delete file \"${SHUTDOWNFLAGFILE}\" to abort shutdown after speed test"
trap 'sleep 120; if [ -f "${SHUTDOWNFLAGFILE}" ]; then rm -f "$
{SHUTDOWNFLAGFILE}"; sudo shutdown now; fi' EXIT

trap 'echo Error "$?" al line ${LINENO} in script ${SCRIPTNAME} ; exit 1' \
    ERR

trap 'echo Terminated al line ${LINENO} in script ${SCRIPTNAME} ; exit 2' \
    HUP INT ABRT KILL
```

Test Script (1)

Main control loop:

```
# execute a sequence of test runs until end or a series of consecutive failures
while [[ "${COUNT}" -gt 0 ]]
do
    # Execute a sequence of downloads until one fails or all succeed
    if download "${URL_1}" && download "${URL_2}" && download "${URL_3}"
    then
        # send any cached data to disk
        sync
        # reset error flag
        ERRORCOUNT=0
    else
        # report download errors
```

Test Script (2)

Principal part of the download function:

```
    else
        # next command is used to provide a timestamp and url in case wget fails
        # wrap the command in a test so that script will not stop on failure
        if echo "TimeStamp $(date '+%Y-%m-%d %H:%M:%S') URL=$URL"; then
true; fi
        # now get the download
        time -p {
            # keep time stamp, url, download size, but discard progress report
            wget ${WGET_OPTIONS} -nv -O /dev/null ${1}
            WGETCODE=$?
        }
        TIMECODE=$?     # should be same as WGETCODE, see time(1)
        sync
```

Test Script (3)

Error Management:

```
# Information for figuring out what went wrong (if anything did)
echo "Script \"${SCRIPTNAME}\" running as USER=\"${USER=}\" with
PATH=\"${PATH=}\""

trap 'echo Error "$?" al line ${LINENO} in script ${SCRIPTNAME} ; exit 1' \
    ERR

trap 'echo Terminated al line ${LINENO} in script ${SCRIPTNAME} ; exit 2' \
    HUP INT ABRT KILL
```

Download File and Report File

Log Entry (5 lines per download - TimeStamp/URL:/real/user/sys)

```
TimeStamp 2016-04-07 17:24:37
URL=http://web.ncf.ca/am125/speedtest/random15megabytes
2016-04-07 17:24:45
URL:http://web.ncf.ca/am125/speedtest/random15megabytes
[15000000/15000000] -> "/dev/null" [1]
real 8.19
user 0.48
sys 1.94
```

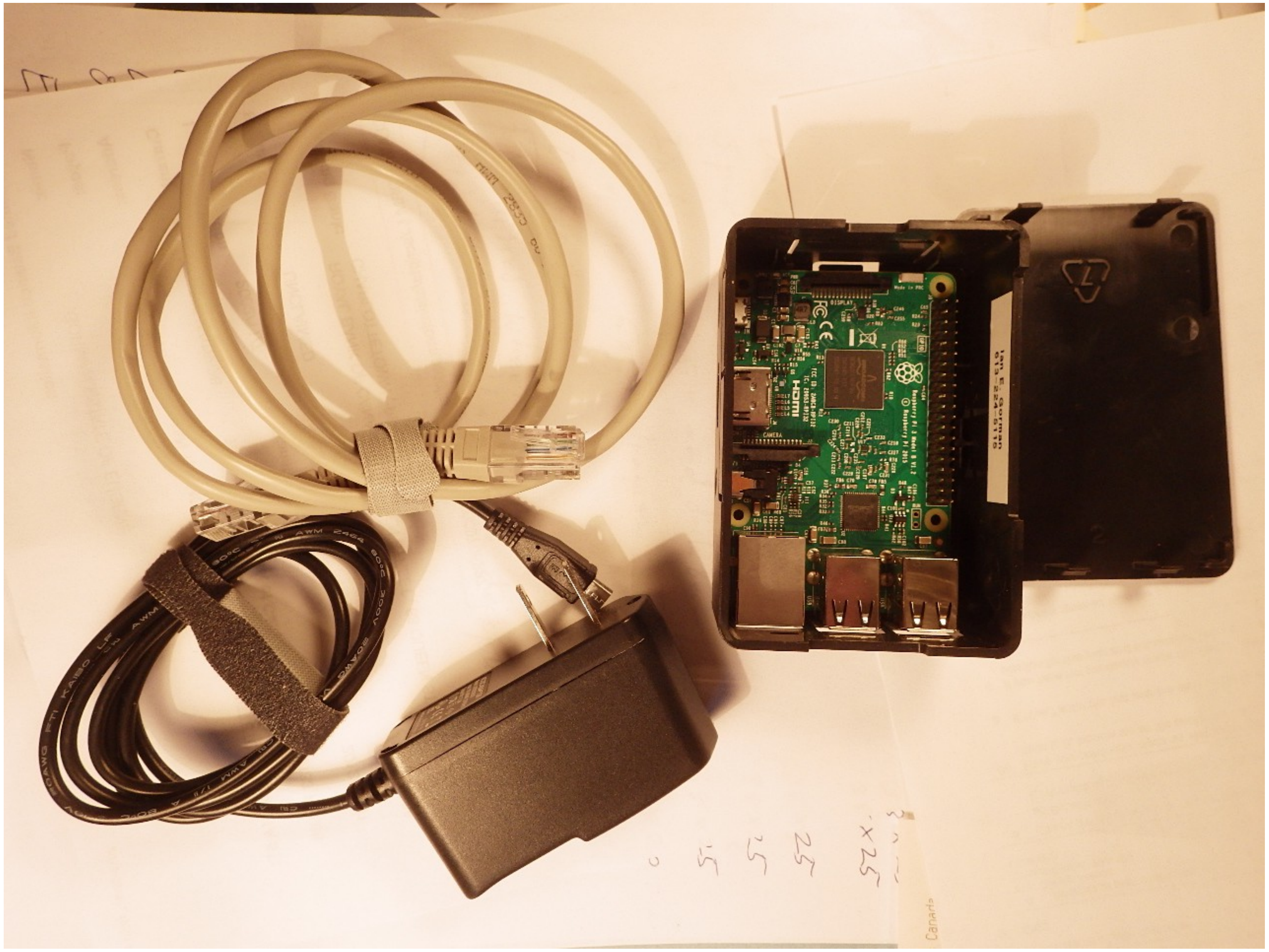
Report Input File (one line per download)

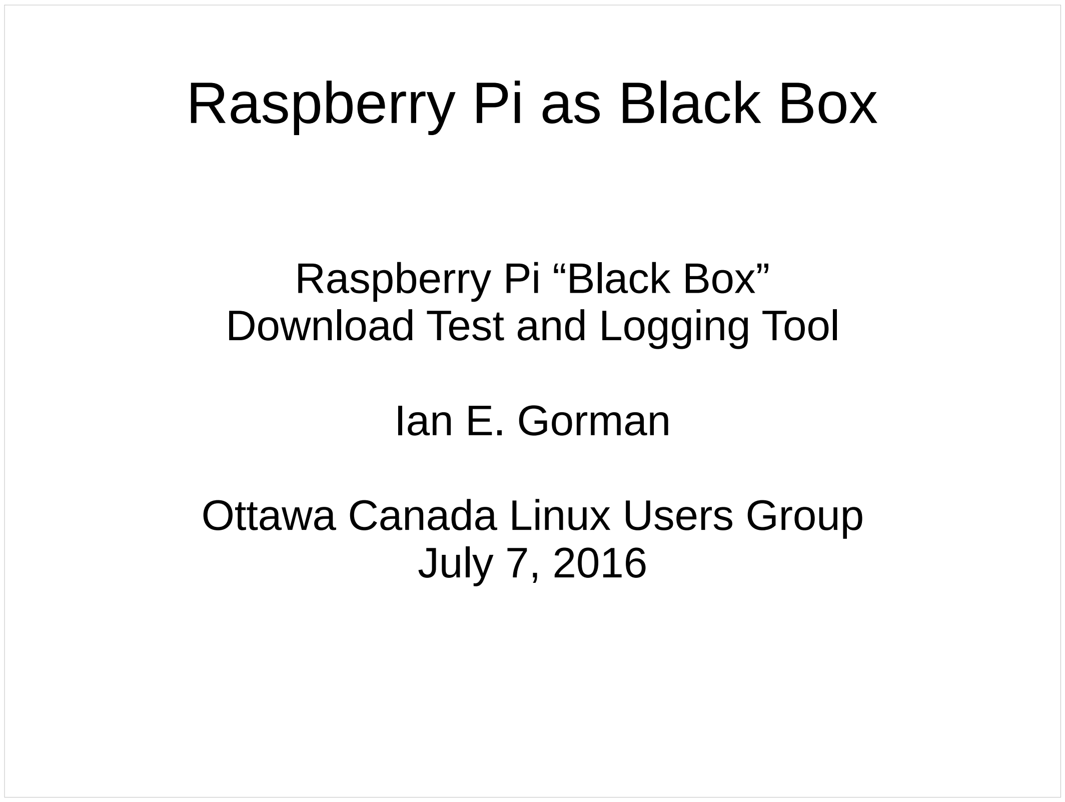
```
date      time      source  bytes    bits seconds      bps host
2016-04-07 17:24:45
http://web.ncf.ca/am125/speedtest/random15megabytes 15000000
120000000  8.19      1.4652e+07 web.ncf.ca
```

AWK Program to Convert Log File

```
# timestamp and URL should be given by 'wget'
# Get the secondary timestamp, url and host to be used when wget fails
/^TimeStamp/    {
    stamp_date = $2
    stamp_time = $3
    sub(/^.*URL=/, "")
    stamp_url = $0
    stamp_host = stamp_url
    sub(/^.*:\/\//, "", stamp_host)   # trim all before hostname
    sub(/\/.*$/, "", stamp_host)      # trim all after hostname
}

# get data from the output of 'wget' and clear the previous time report
/^2[0-9]/  {    # Pick up the year, but with a Y3K problem  ;)
    …
}

# get data from the following 'time' report and print a one-line test summary
/^ *real/  {   # pick up the "wall clock" line from the 'time' report
    …
}
```

The Black Box Test Kit



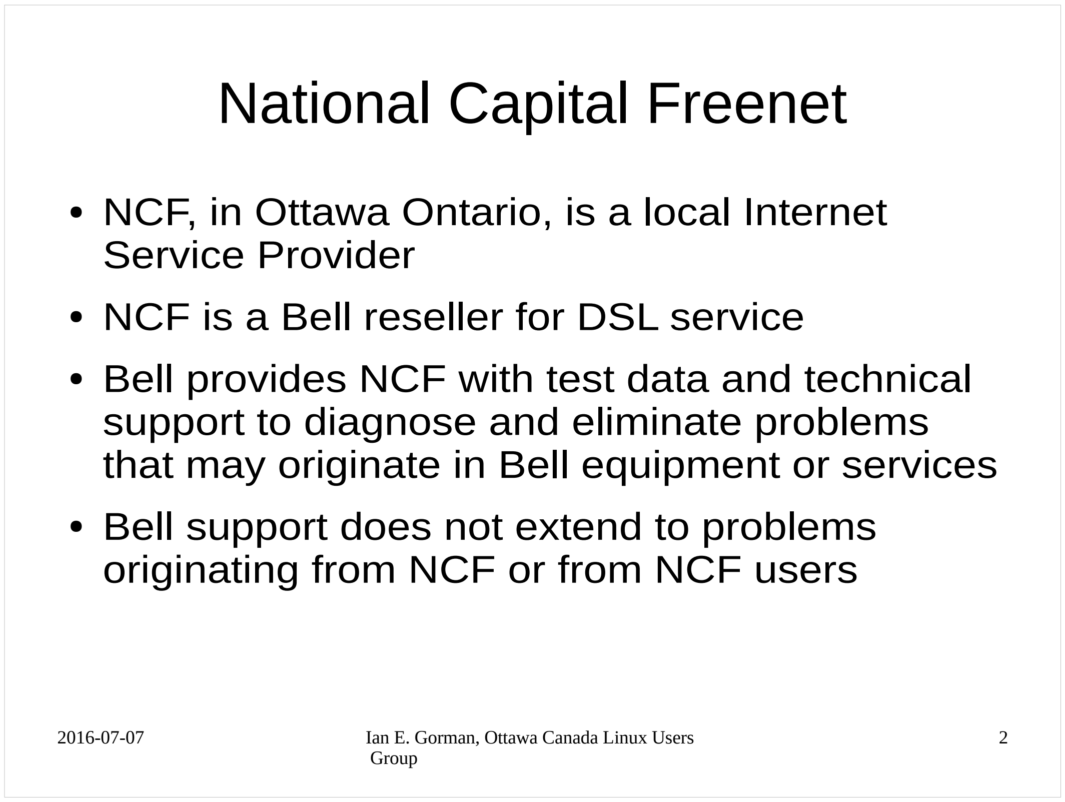


Raspberry Pi as Black Box

Raspberry Pi “Black Box”
Download Test and Logging Tool

Ian E. Gorman

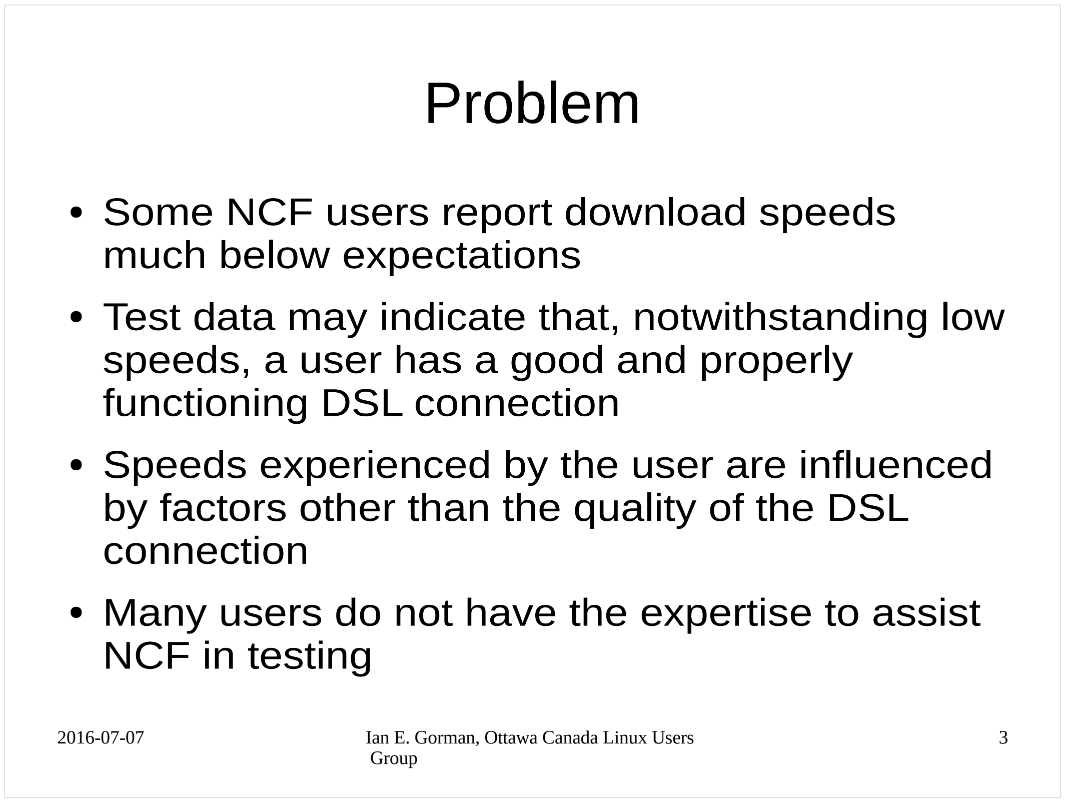
Ottawa Canada Linux Users Group
July 7, 2016



National Capital Freenet

- NCF, in Ottawa Ontario, is a local Internet Service Provider
- NCF is a Bell reseller for DSL service
- Bell provides NCF with test data and technical support to diagnose and eliminate problems that may originate in Bell equipment or services
- Bell support does not extend to problems originating from NCF or from NCF users

2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 2



Problem

- Some NCF users report download speeds much below expectations
- Test data may indicate that, notwithstanding low speeds, a user has a good and properly functioning DSL connection
- Speeds experienced by the user are influenced by factors other than the quality of the DSL connection
- Many users do not have the expertise to assist NCF in testing

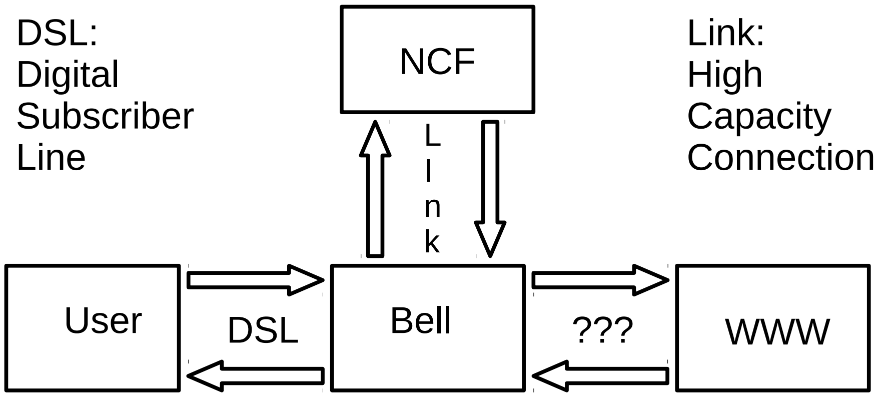
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 3

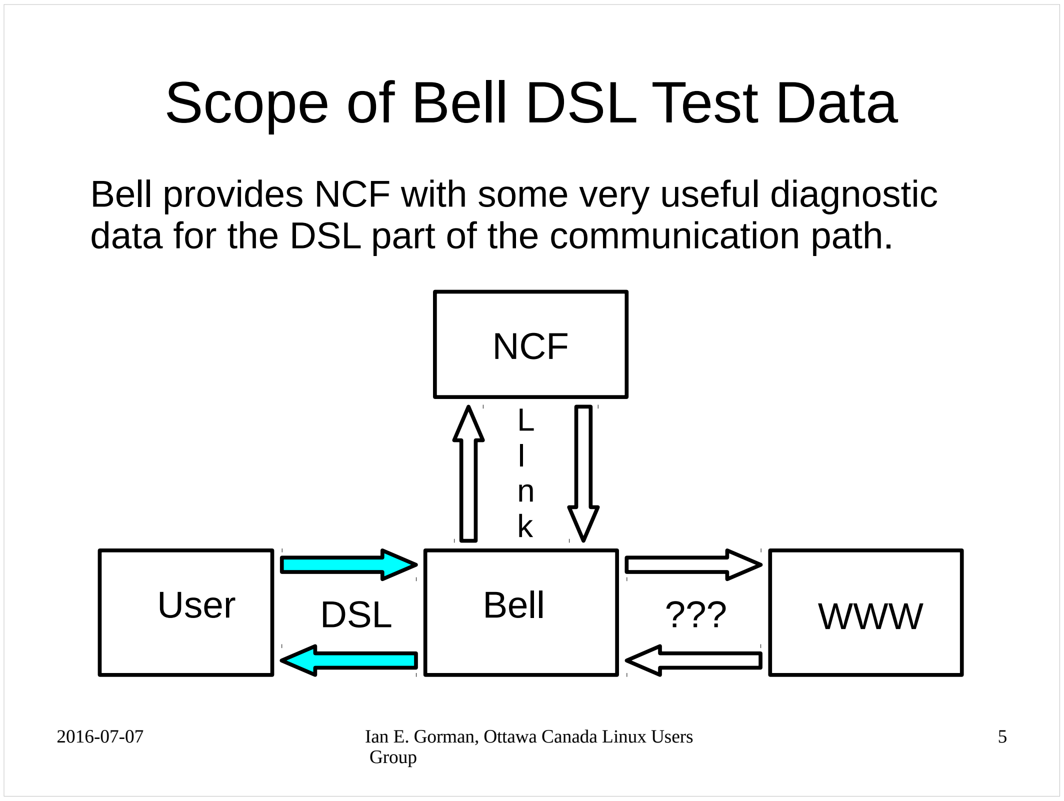
NCF Connection to WWW

Bell provides communications infrastructure, used by NCF to provide Internet services to end user.

DSL:
Digital
Subscriber
Line

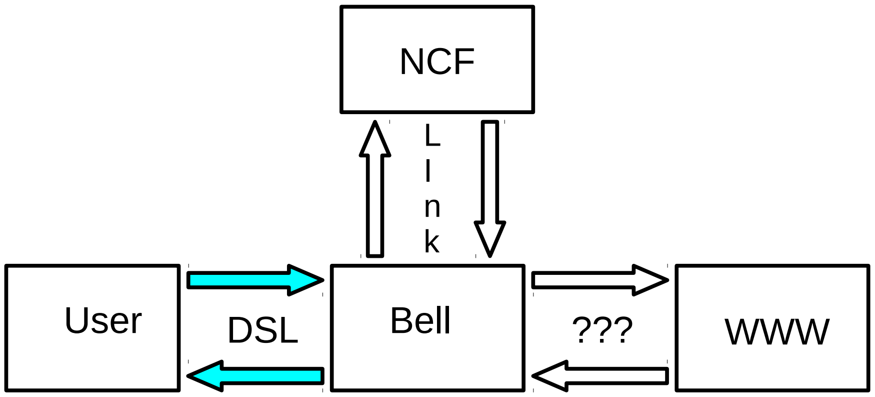
Link:
High
Capacity
Connection



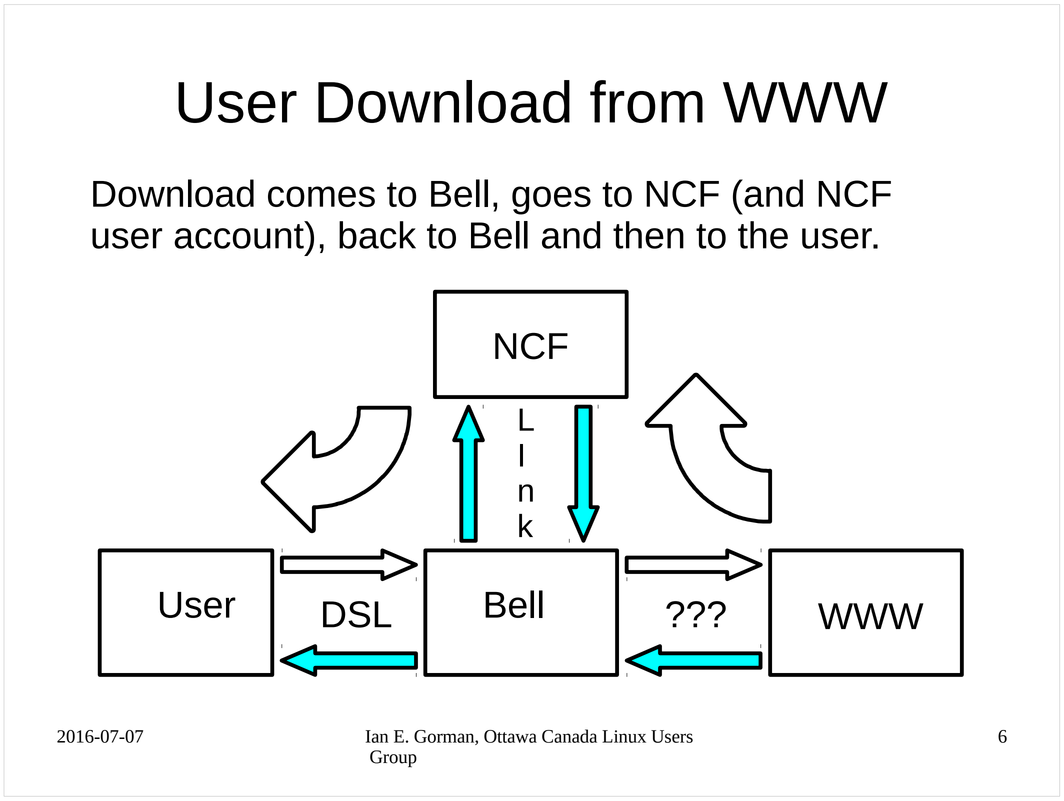


Scope of Bell DSL Test Data

Bell provides NCF with some very useful diagnostic data for the DSL part of the communication path.

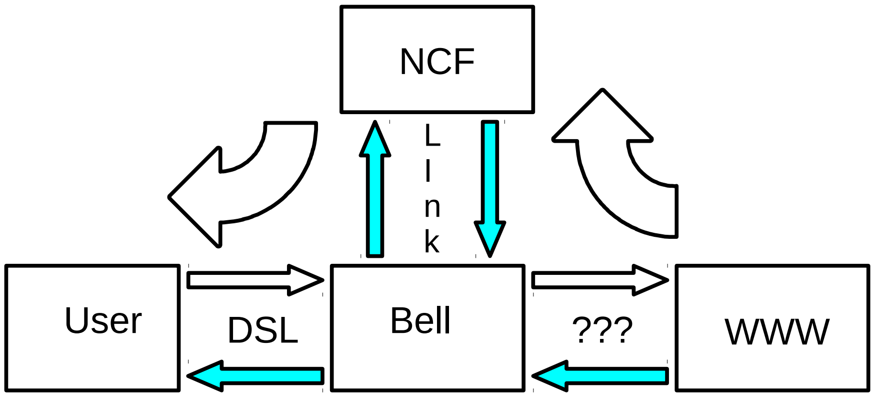


2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 5

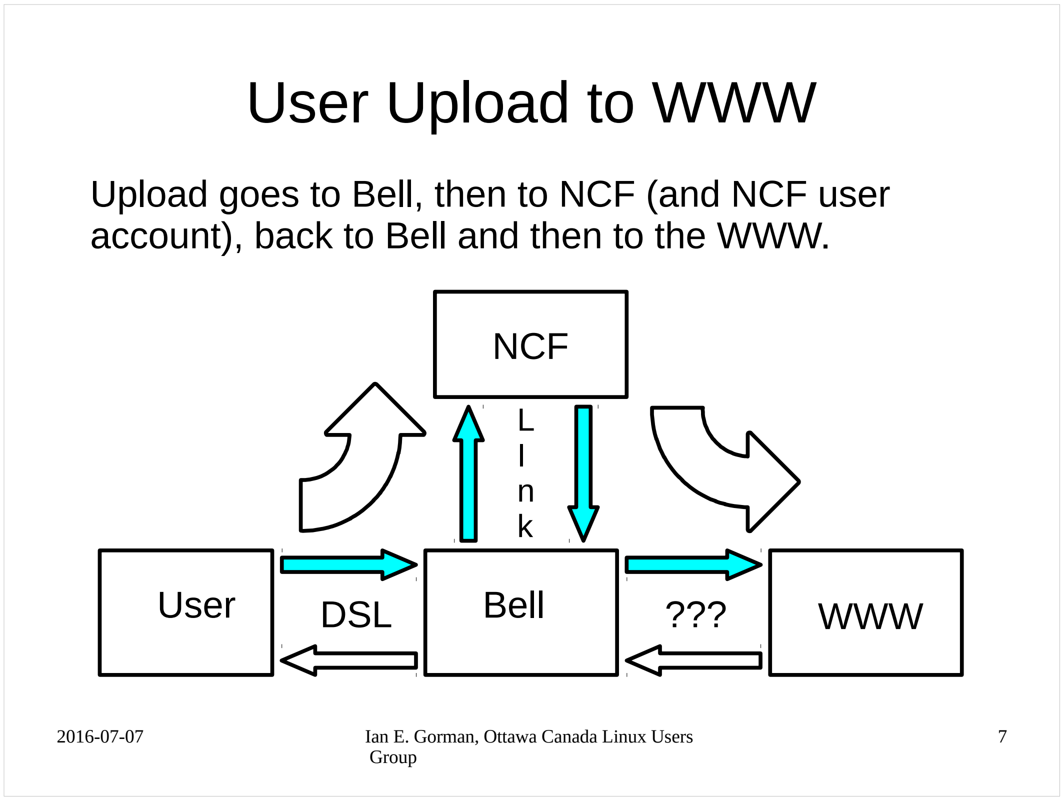


User Download from WWW

Download comes to Bell, goes to NCF (and NCF user account), back to Bell and then to the user.

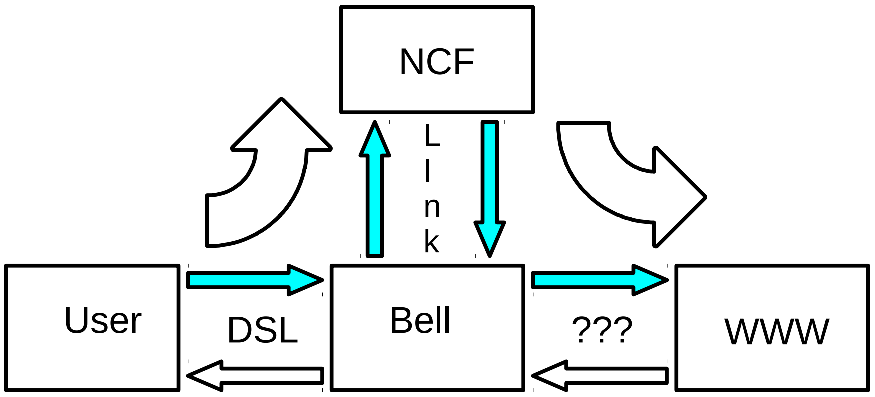


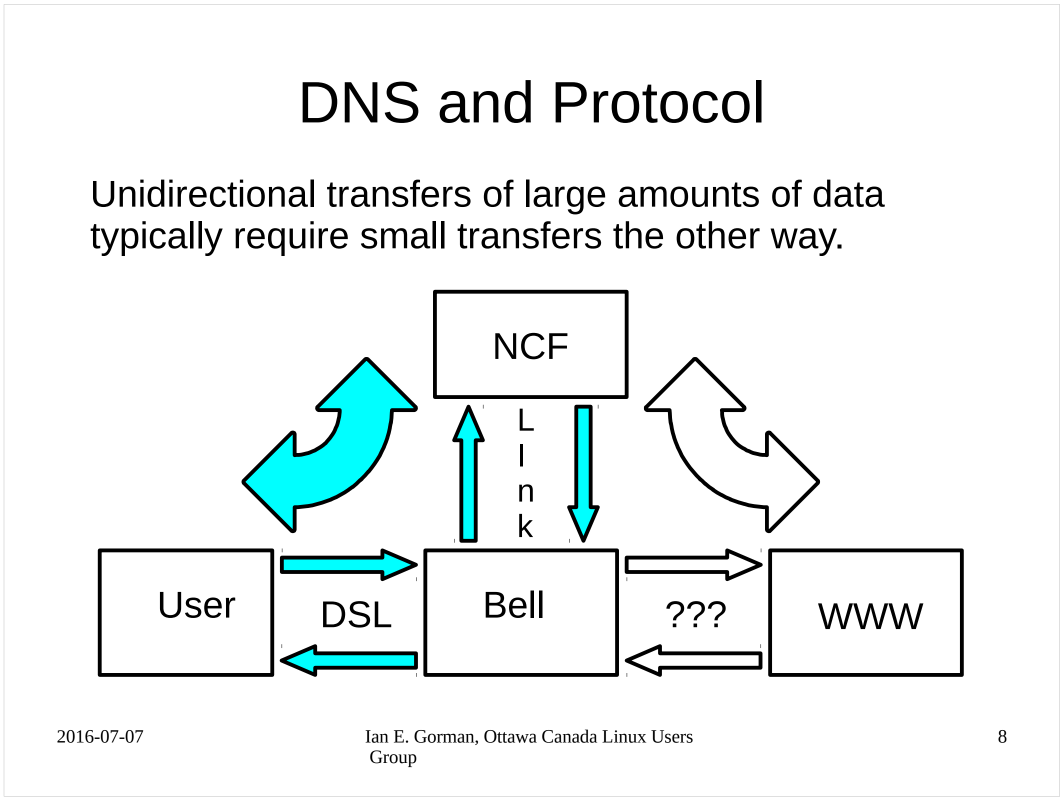
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 6



User Upload to WWW

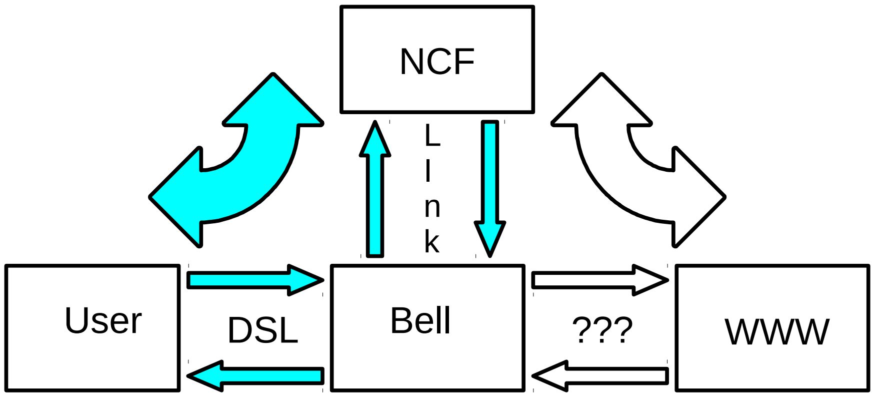
Upload goes to Bell, then to NCF (and NCF user account), back to Bell and then to the WWW.



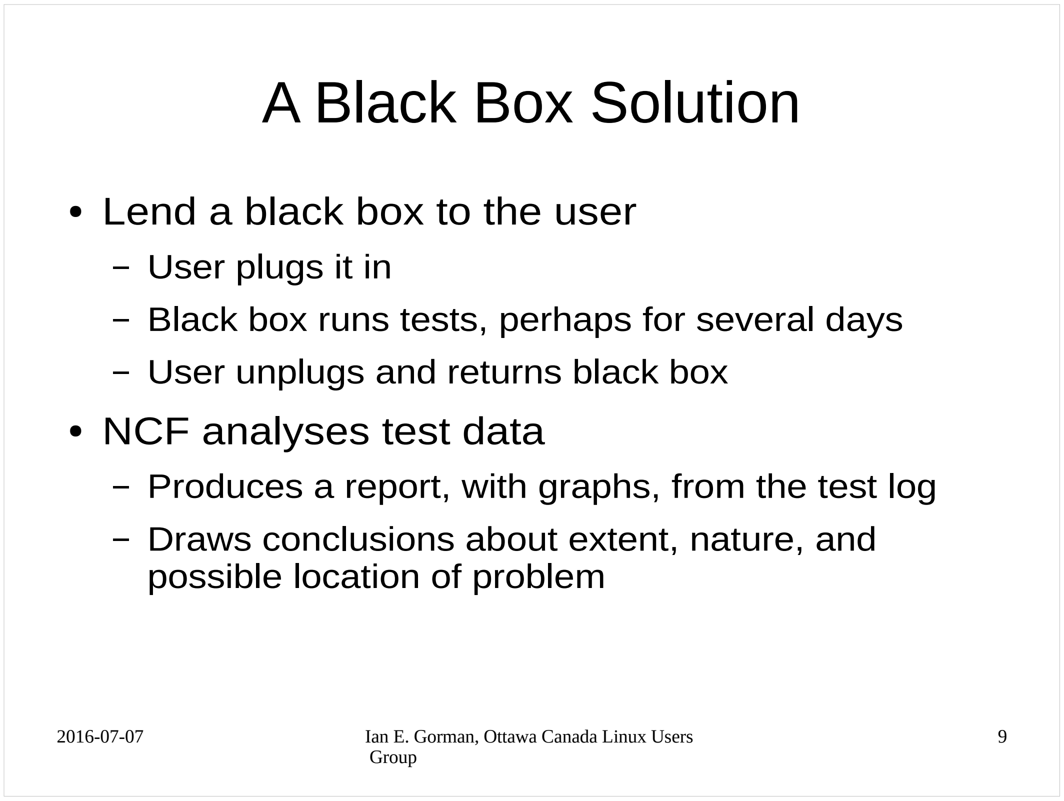


DNS and Protocol

Unidirectional transfers of large amounts of data typically require small transfers the other way.



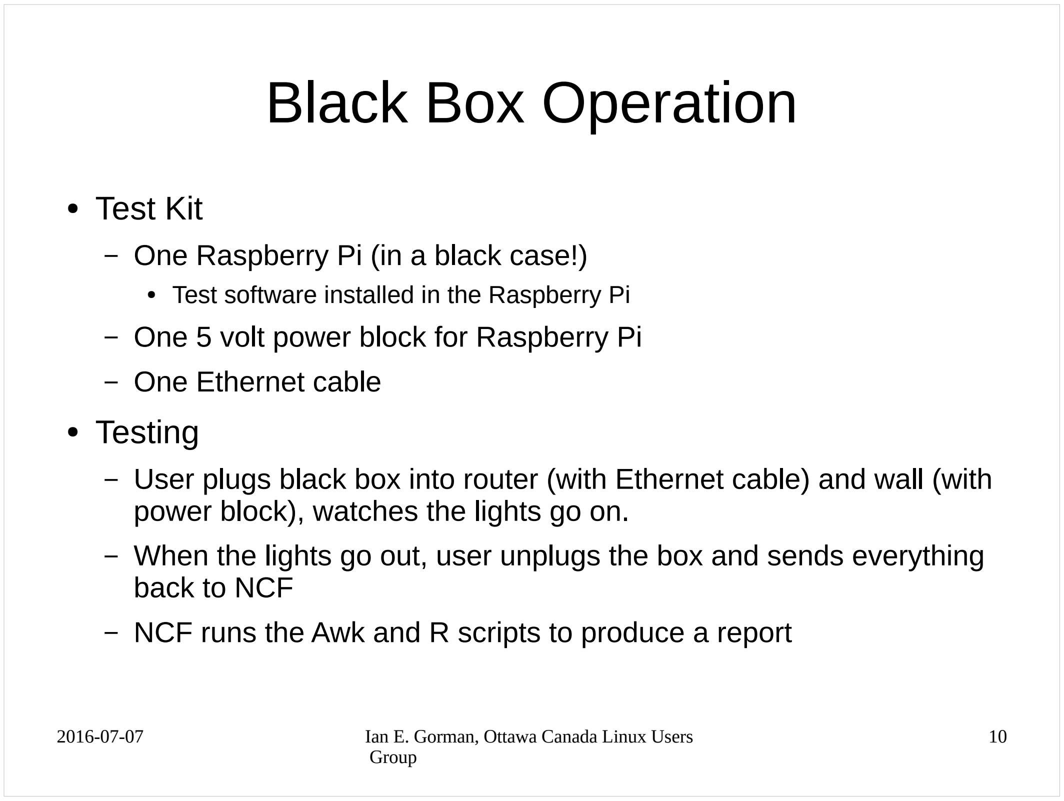
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 8



A Black Box Solution

- Lend a black box to the user
  - User plugs it in
  - Black box runs tests, perhaps for several days
  - User unplugs and returns black box
- NCF analyses test data
  - Produces a report, with graphs, from the test log
  - Draws conclusions about extent, nature, and possible location of problem

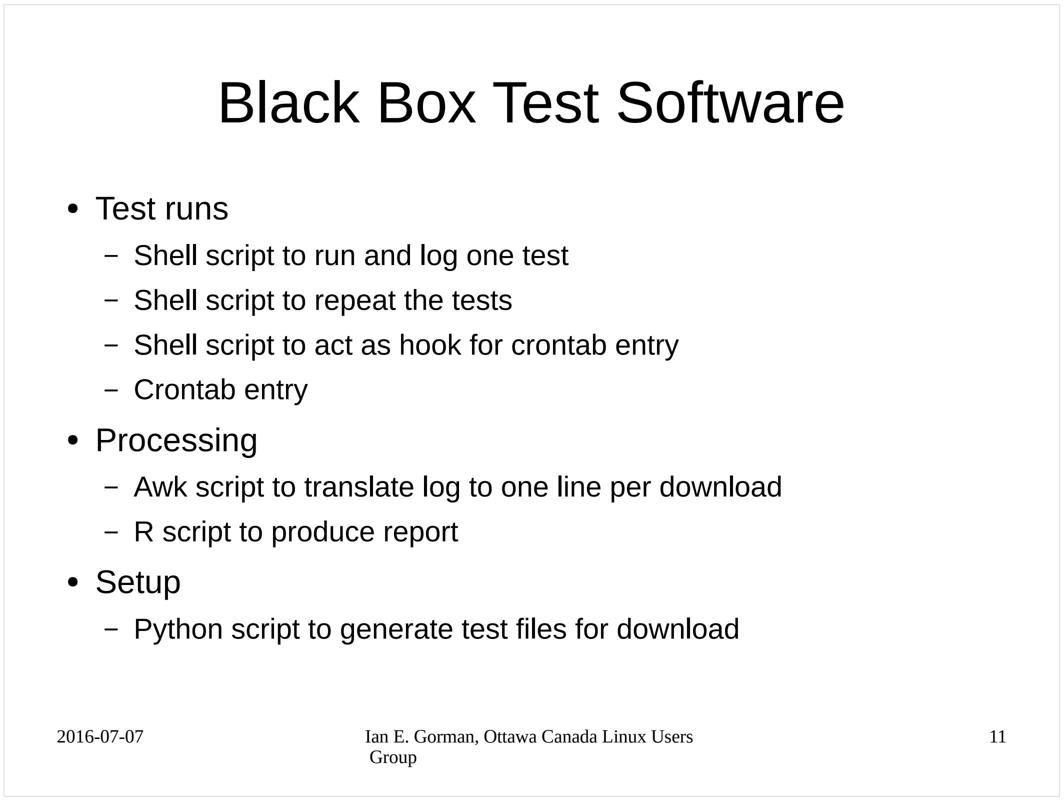
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 9



Black Box Operation

- Test Kit
  - One Raspberry Pi (in a black case!)
    - Test software installed in the Raspberry Pi
  - One 5 volt power block for Raspberry Pi
  - One Ethernet cable
- Testing
  - User plugs black box into router (with Ethernet cable) and wall (with power block), watches the lights go on.
  - When the lights go out, user unplugs the box and sends everything back to NCF
  - NCF runs the Awk and R scripts to produce a report

2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 10

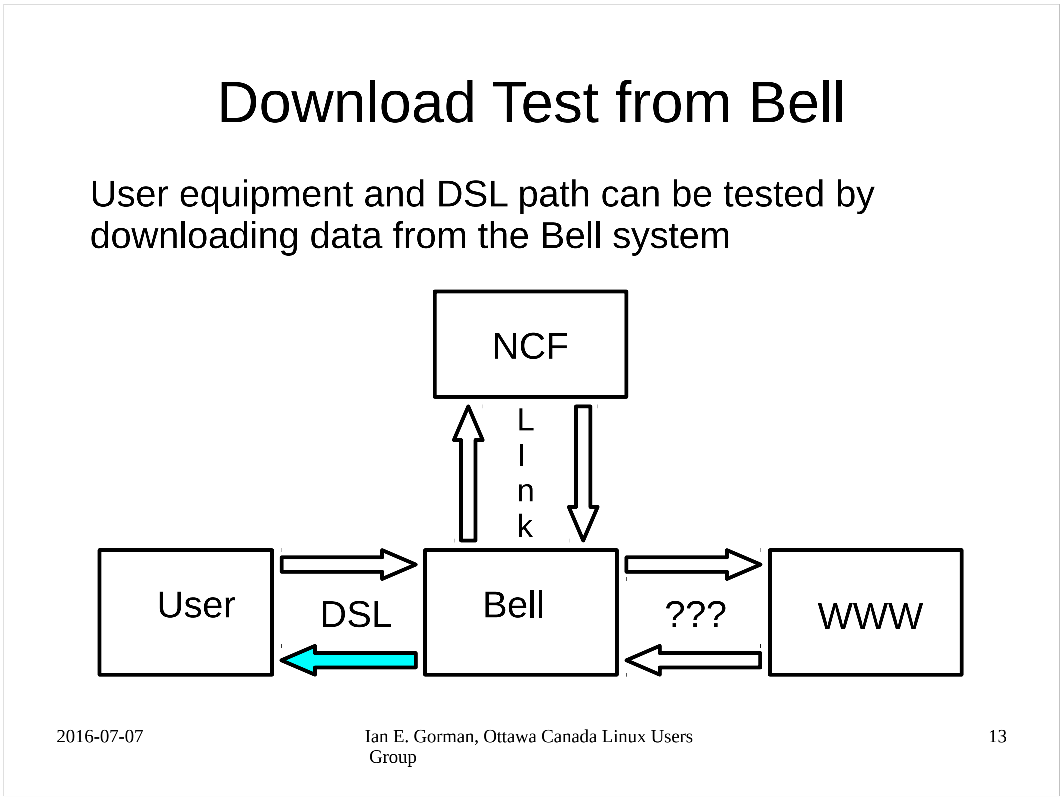


Black Box Test Software

- Test runs
  - Shell script to run and log one test
  - Shell script to repeat the tests
  - Shell script to act as hook for crontab entry
  - Crontab entry
- Processing
  - Awk script to translate log to one line per download
  - R script to produce report
- Setup
  - Python script to generate test files for download

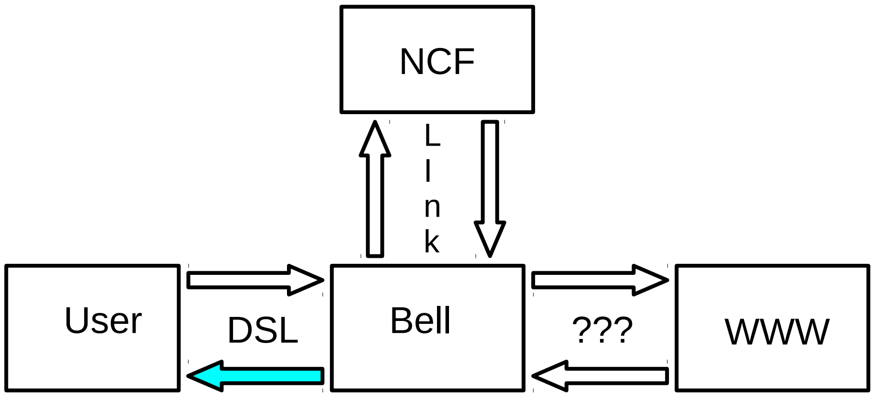
Progressive Test Strategy

- Link (from Bell data)
- DSL (from Bell data)
- User Equipment + DSL
  - User modem is tested, user takes black box home
  - Black box records downloads from Bell
- User Equipment + DSL + Link + NCF
  - Black box records downloads from NCF
- User Equipment + DSL + Link + NCF + WWW
  - Black box records downloads from WWW



Download Test from Bell

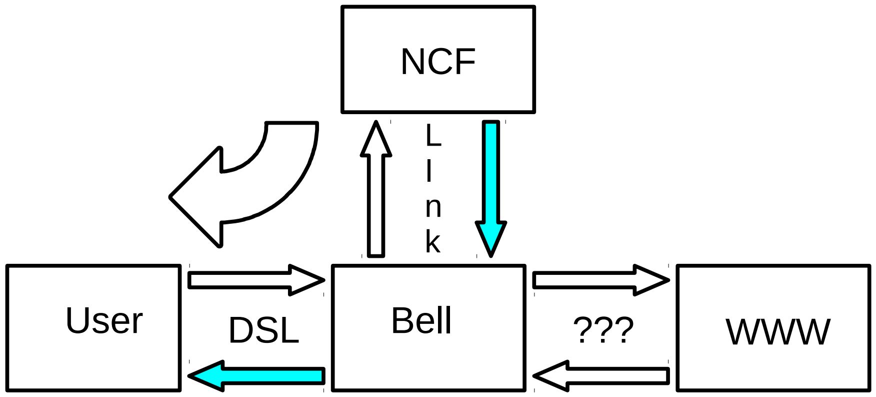
User equipment and DSL path can be tested by downloading data from the Bell system

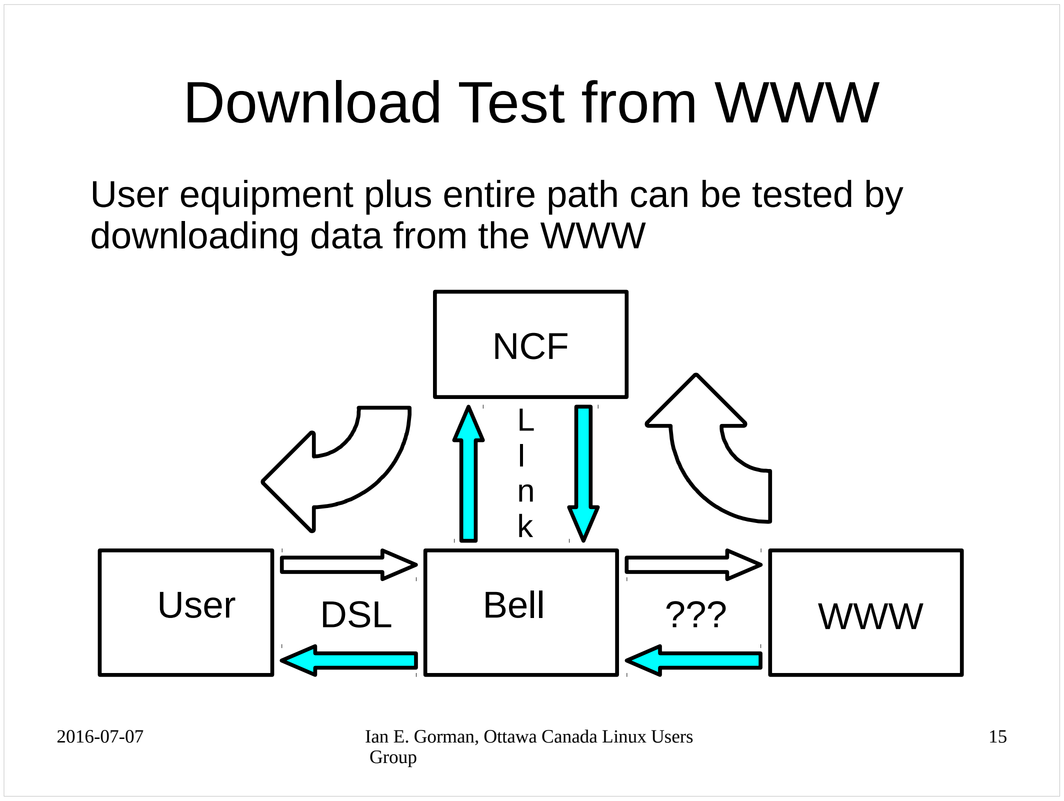


2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 13

Download Test from NCF

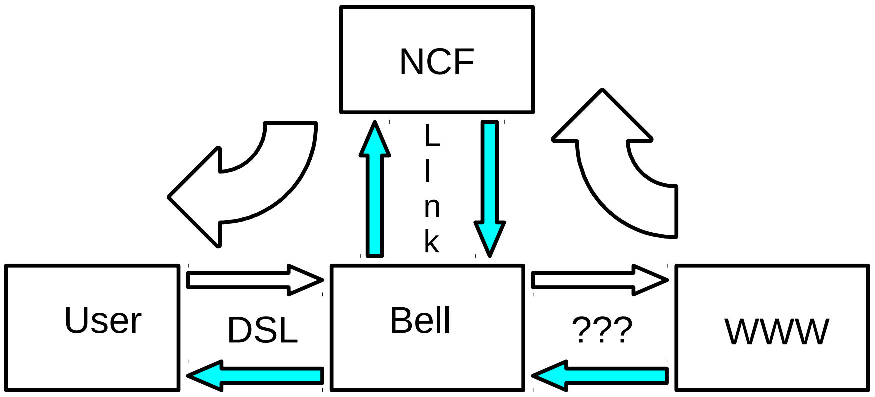
User equipment, DSL path, and Link path can be tested by downloading data from the NCF system





Download Test from WWW

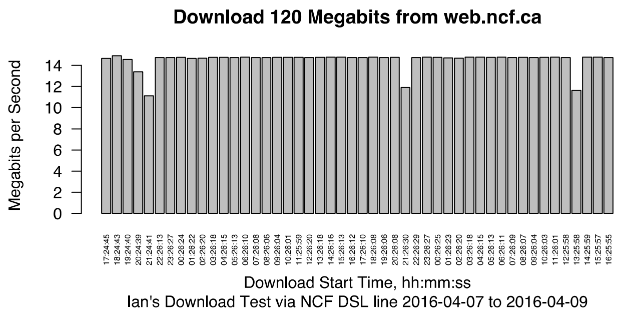
User equipment plus entire path can be tested by downloading data from the WWW

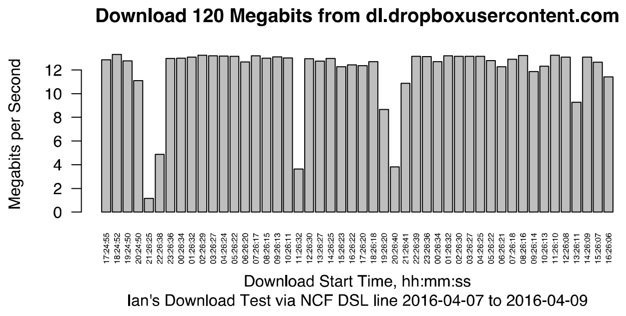


2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 15

Bar Chart

Compare speeds from up to three download sources

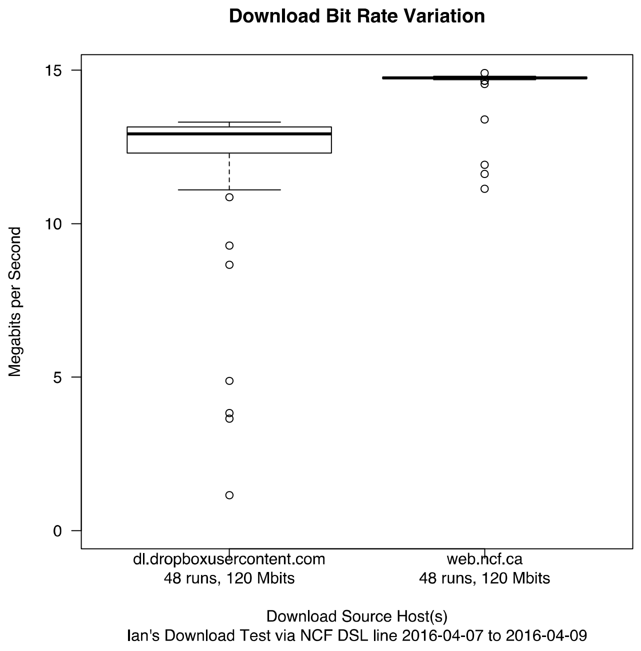


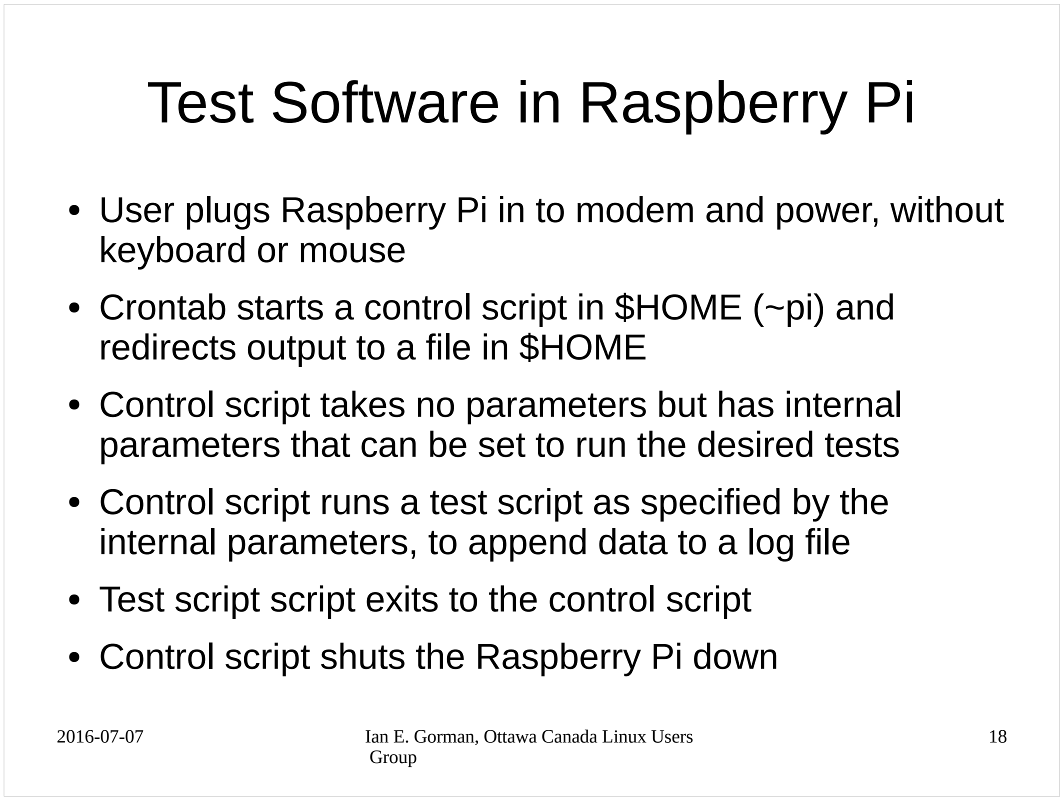


2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 16

Scatter Chart

Compare variation from up to three download sources

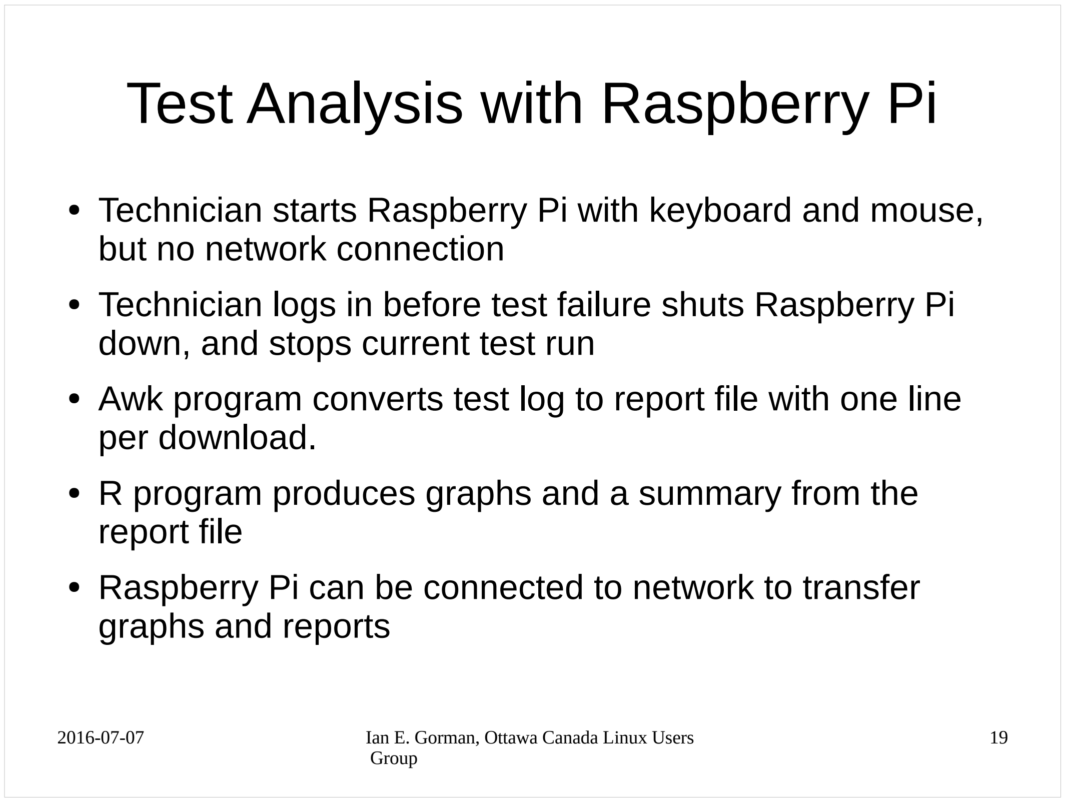




Test Software in Raspberry Pi

- User plugs Raspberry Pi in to modem and power, without keyboard or mouse
- Crontab starts a control script in $HOME (~pi) and redirects output to a file in $HOME
- Control script takes no parameters but has internal parameters that can be set to run the desired tests
- Control script runs a test script as specified by the internal parameters, to append data to a log file
- Test script script exits to the control script
- Control script shuts the Raspberry Pi down

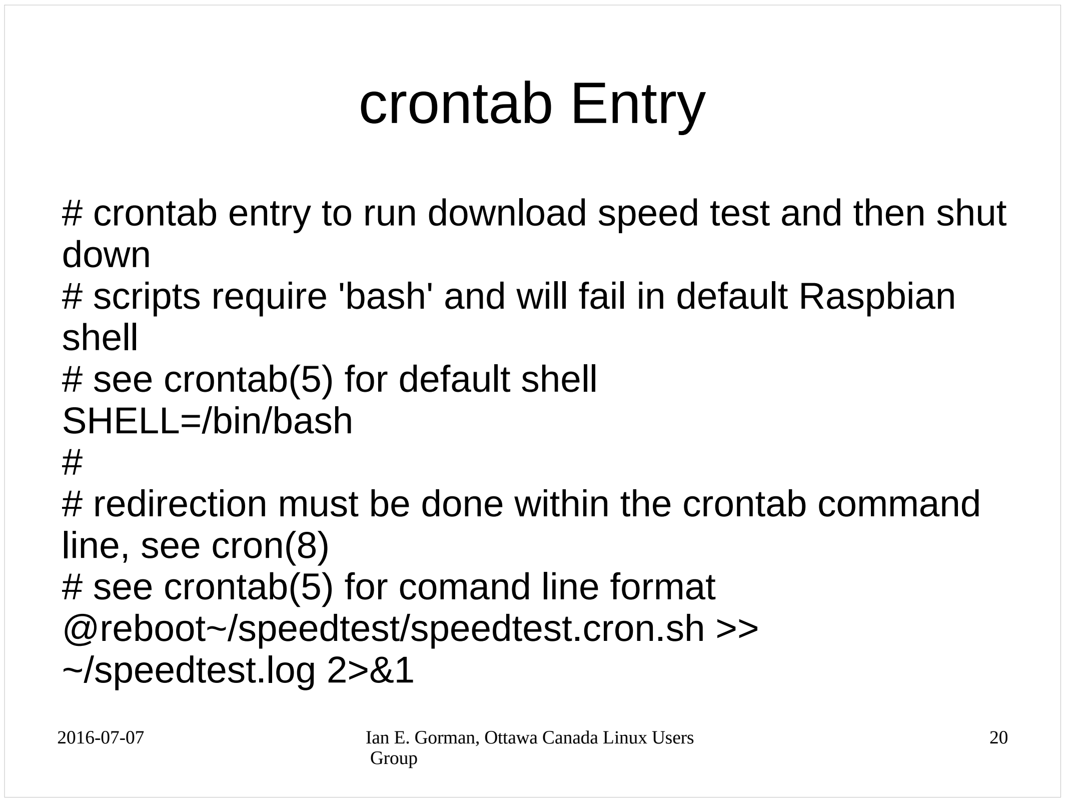
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 18



Test Analysis with Raspberry Pi

- Technician starts Raspberry Pi with keyboard and mouse, but no network connection
- Technician logs in before test failure shuts Raspberry Pi down, and stops current test run
- Awk program converts test log to report file with one line per download.
- R program produces graphs and a summary from the report file
- Raspberry Pi can be connected to network to transfer graphs and reports

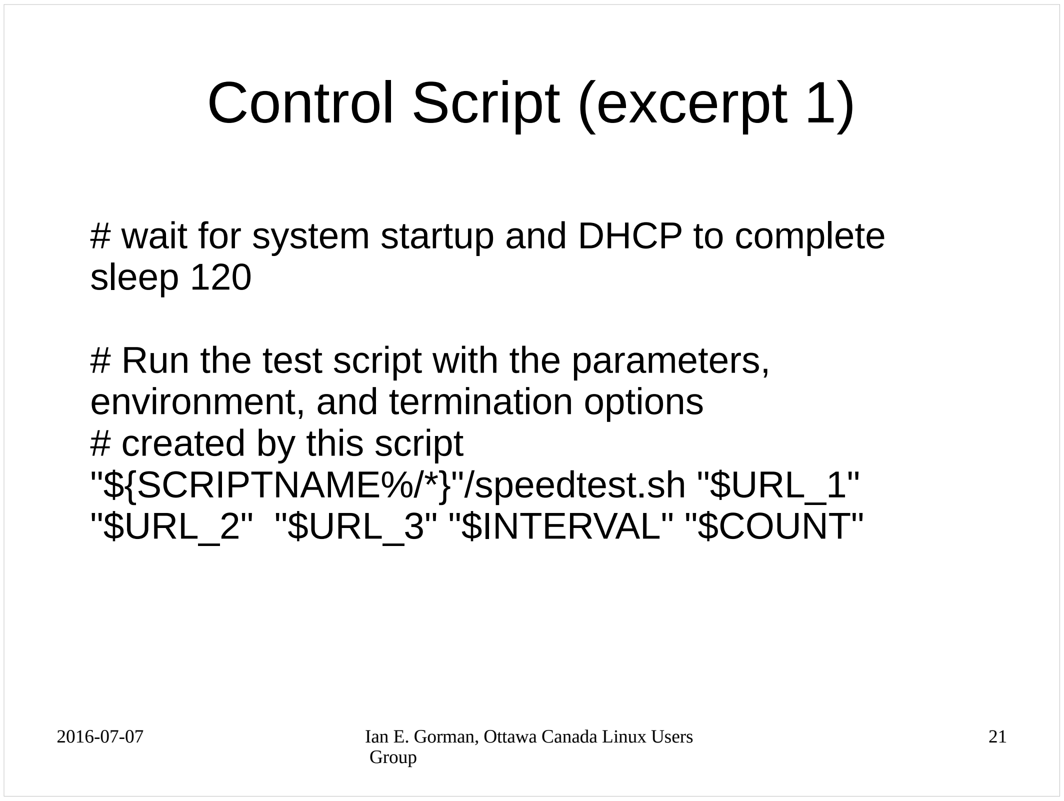
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 19



crontab Entry

```
# crontab entry to run download speed test and then shut
down
# scripts require 'bash' and will fail in default Raspbian
shell
# see crontab(5) for default shell
SHELL=/bin/bash
#
# redirection must be done within the crontab command
line, see cron(8)
# see crontab(5) for comand line format
@reboot~/speedtest/speedtest.cron.sh >>
~/speedtest.log 2>&1
```

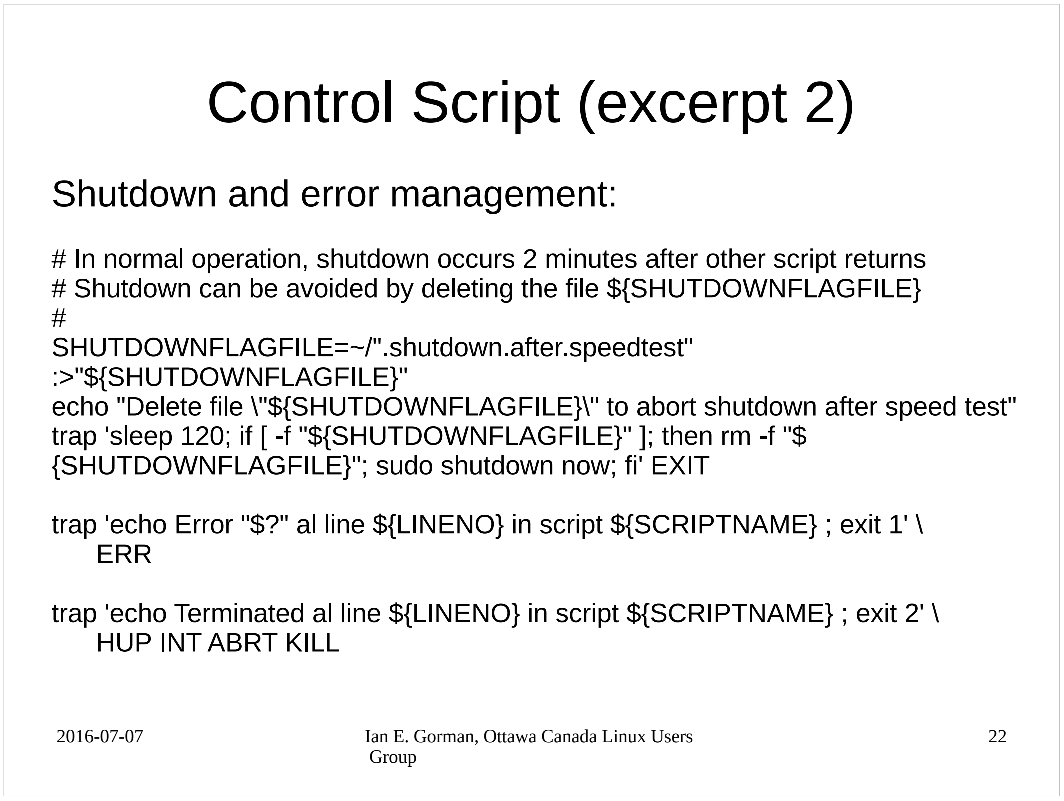
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 20



Control Script (excerpt 1)

```
# wait for system startup and DHCP to complete
sleep 120

# Run the test script with the parameters,
environment, and termination options
# created by this script
"${SCRIPTNAME%/*}"/speedtest.sh "$URL_1"
"$URL_2"  "$URL_3" "$INTERVAL" "$COUNT"
```

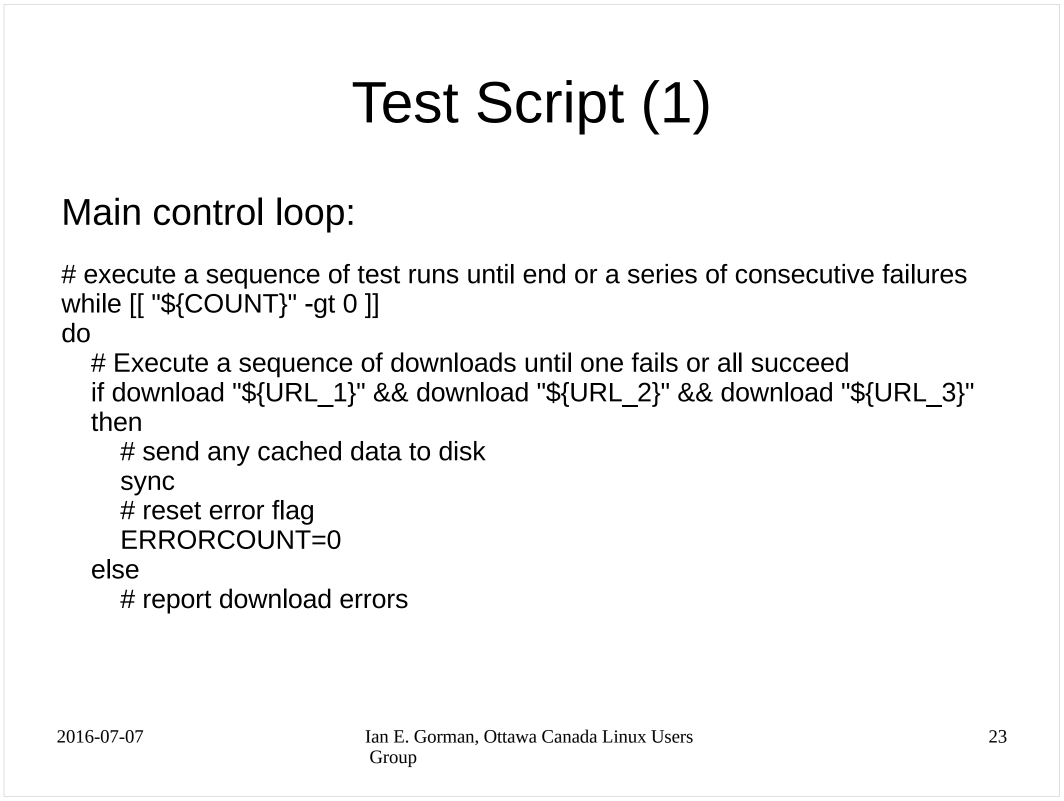
2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 21



Control Script (excerpt 2)

Shutdown and error management:

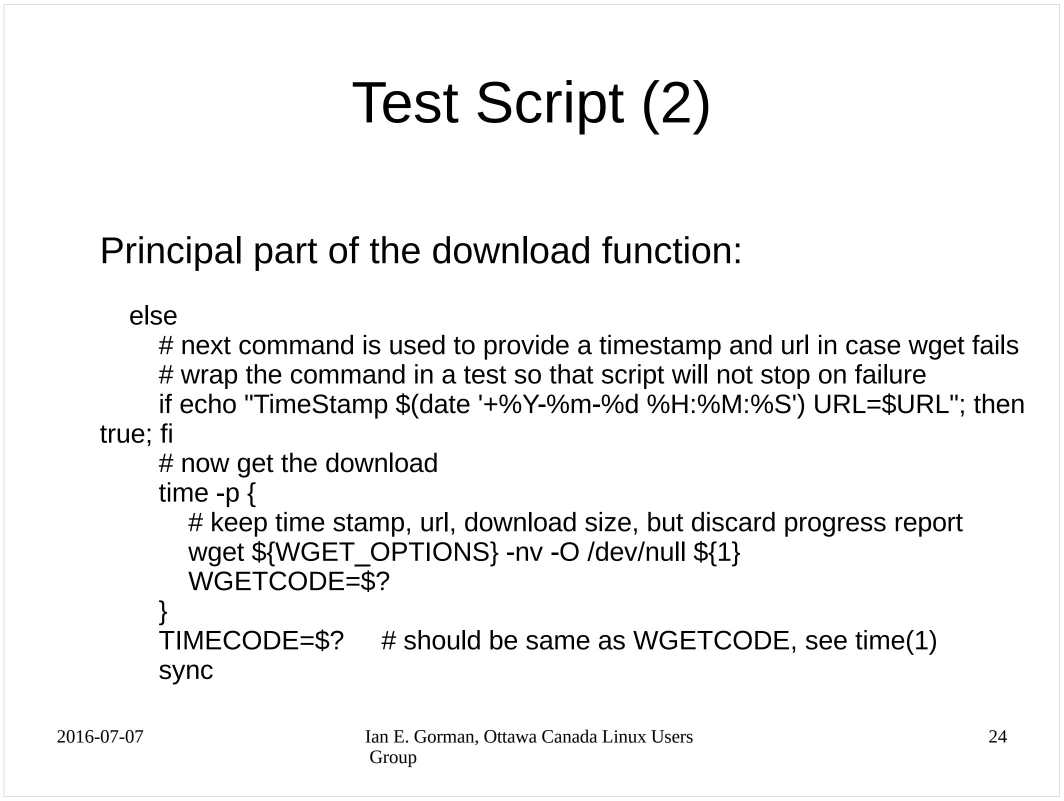
```
# In normal operation, shutdown occurs 2 minutes after other script returns
# Shutdown can be avoided by deleting the file ${SHUTDOWNFLAGFILE}
#
SHUTDOWNFLAGFILE=~/".shutdown.after.speedtest"
:>"${SHUTDOWNFLAGFILE}"
echo "Delete file \"${SHUTDOWNFLAGFILE}\" to abort shutdown after speed test"
trap 'sleep 120; if [ -f "${SHUTDOWNFLAGFILE}" ]; then rm -f "$
{SHUTDOWNFLAGFILE}"; sudo shutdown now; fi' EXIT

trap 'echo Error "$?" al line ${LINENO} in script ${SCRIPTNAME} ; exit 1' \
    ERR

trap 'echo Terminated al line ${LINENO} in script ${SCRIPTNAME} ; exit 2' \
    HUP INT ABRT KILL
```



Test Script (1)

Main control loop:

```
# execute a sequence of test runs until end or a series of consecutive failures
while [[ "${COUNT}" -gt 0 ]]
do
    # Execute a sequence of downloads until one fails or all succeed
    if download "${URL_1}" && download "${URL_2}" && download "${URL_3}"
    then
        # send any cached data to disk
        sync
        # reset error flag
        ERRORCOUNT=0
    else
        # report download errors
```

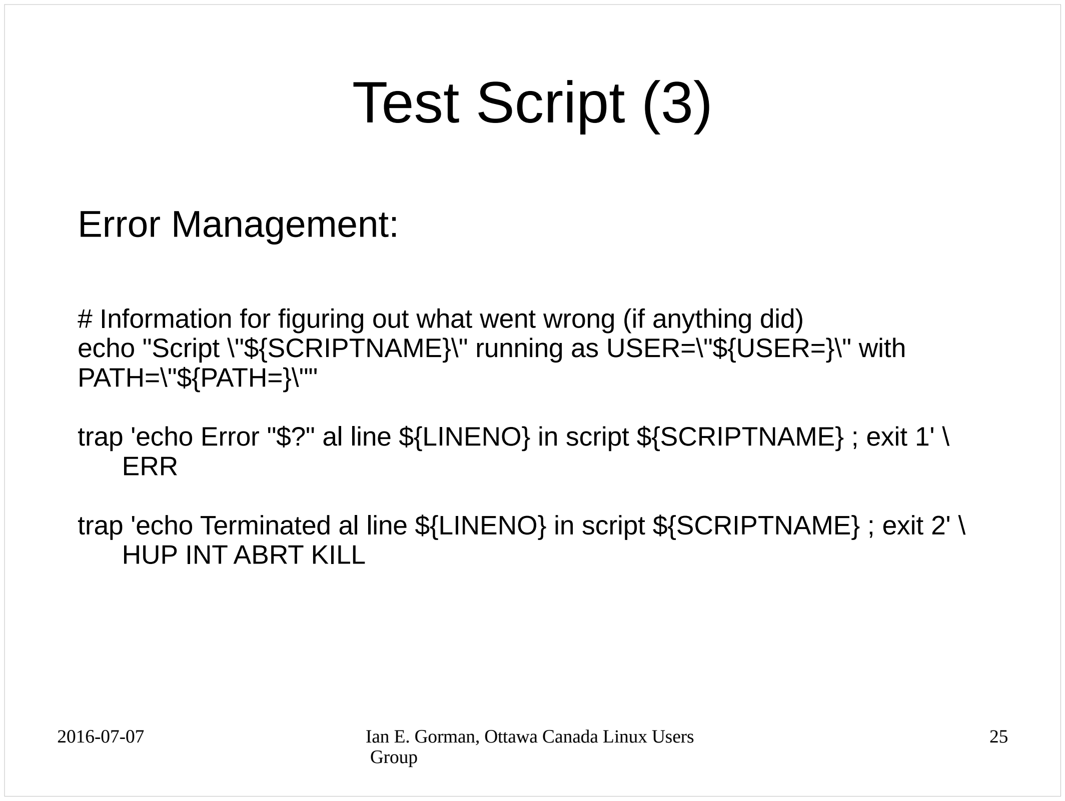


Test Script (2)

Principal part of the download function:

```
  else
      # next command is used to provide a timestamp and url in case wget fails
      # wrap the command in a test so that script will not stop on failure
      if echo "TimeStamp $(date '+%Y-%m-%d %H:%M:%S') URL=$URL"; then true; fi
      # now get the download
      time -p {
          # keep time stamp, url, download size, but discard progress report
          wget ${WGET_OPTIONS} -nv -O /dev/null ${1}
          WGETCODE=$?
      }
      TIMECODE=$?     # should be same as WGETCODE, see time(1)
      sync
```

2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 24

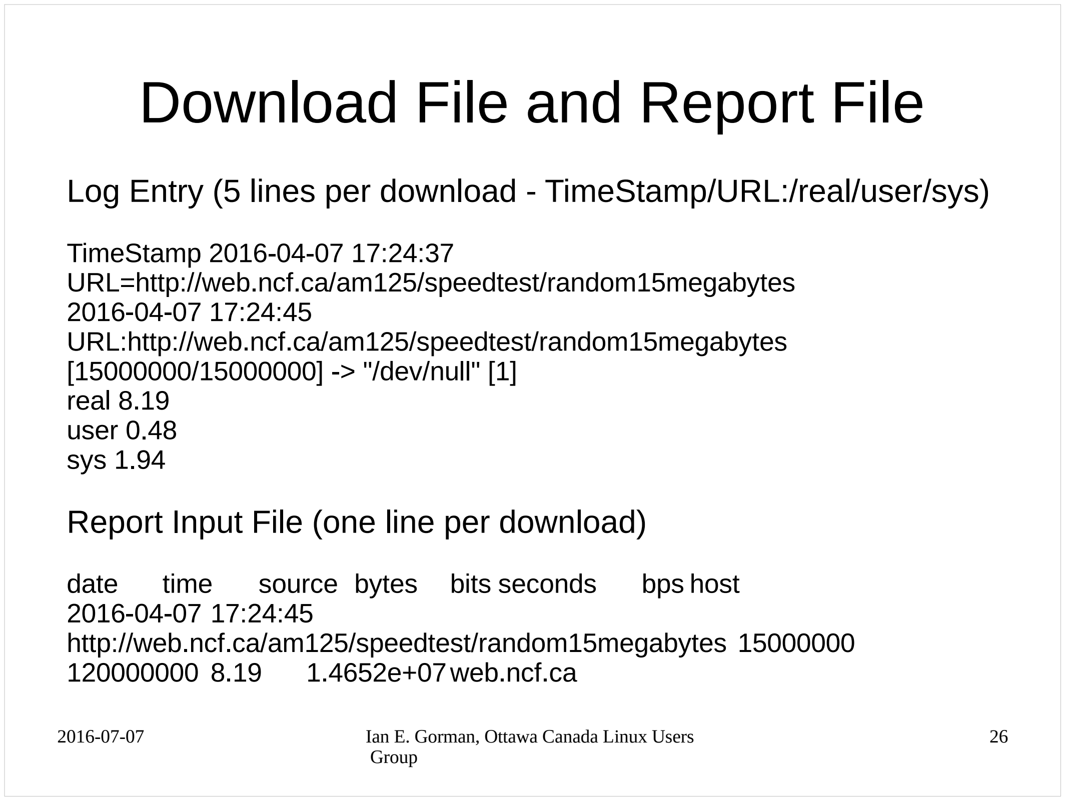


Test Script (3)

Error Management:

```
# Information for figuring out what went wrong (if anything did)
echo "Script \"${SCRIPTNAME}\" running as USER=\"${USER=}\" with
PATH=\"${PATH=}\""

trap 'echo Error "$?" al line ${LINENO} in script ${SCRIPTNAME} ; exit 1' \
      ERR

trap 'echo Terminated al line ${LINENO} in script ${SCRIPTNAME} ; exit 2' \
      HUP INT ABRT KILL
```

2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 25



Download File and Report File

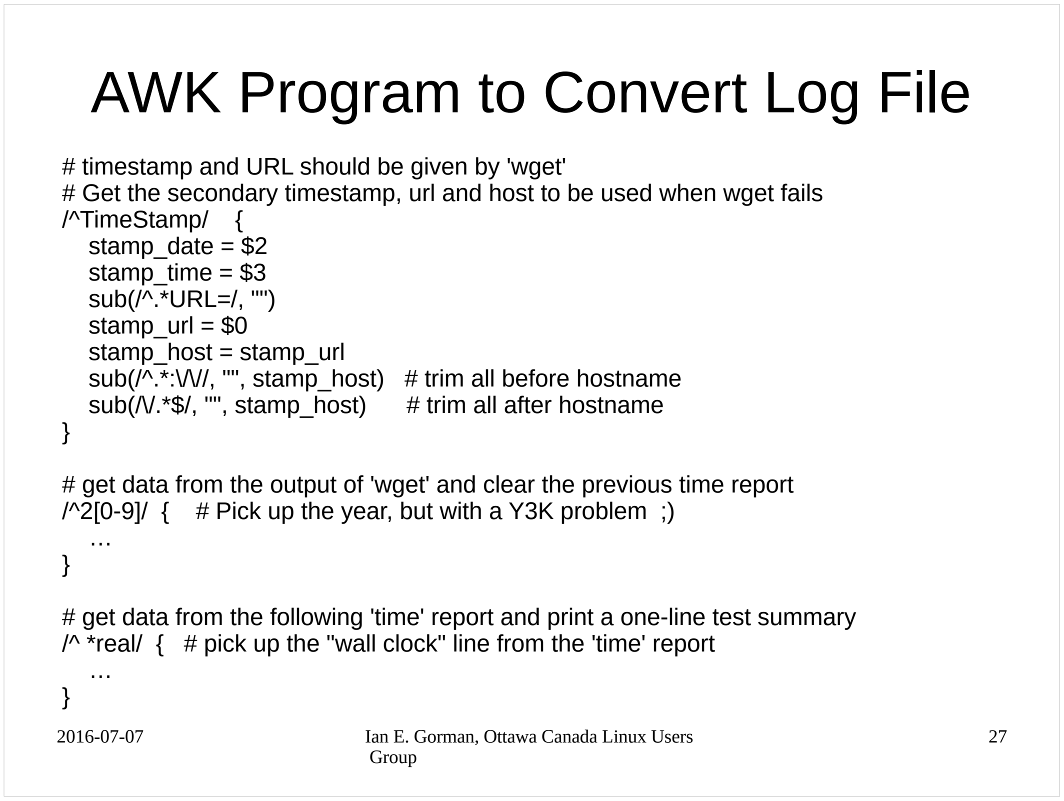
Log Entry (5 lines per download - TimeStamp/URL:/real/user/sys)

```
TimeStamp 2016-04-07 17:24:37
URL=http://web.ncf.ca/am125/speedtest/random15megabytes
2016-04-07 17:24:45
URL:http://web.ncf.ca/am125/speedtest/random15megabytes
[15000000/15000000] -> "/dev/null" [1]
real 8.19
user 0.48
sys 1.94
```

Report Input File (one line per download)

```
date      time      source  bytes     bits seconds      bps host
2016-04-07 17:24:45
http://web.ncf.ca/am125/speedtest/random15megabytes  15000000
120000000  8.19      1.4652e+07 web.ncf.ca
```

2016-07-07 Ian E. Gorman, Ottawa Canada Linux Users Group 26



AWK Program to Convert Log File

```
# timestamp and URL should be given by 'wget'
# Get the secondary timestamp, url and host to be used when wget fails
/^TimeStamp/    {
    stamp_date = $2
    stamp_time = $3
    sub(/^.*URL=/, "")
    stamp_url = $0
    stamp_host = stamp_url
    sub(/^.*:\/\//, "", stamp_host)   # trim all before hostname
    sub(/\/.*$/, "", stamp_host)      # trim all after hostname
}

# get data from the output of 'wget' and clear the previous time report
/^2[0-9]/  {    # Pick up the year, but with a Y3K problem  ;)
    …
}

# get data from the following 'time' report and print a one-line test summary
/^ *real/  {   # pick up the "wall clock" line from the 'time' report
    …
}
```

The Black Box Test Kit