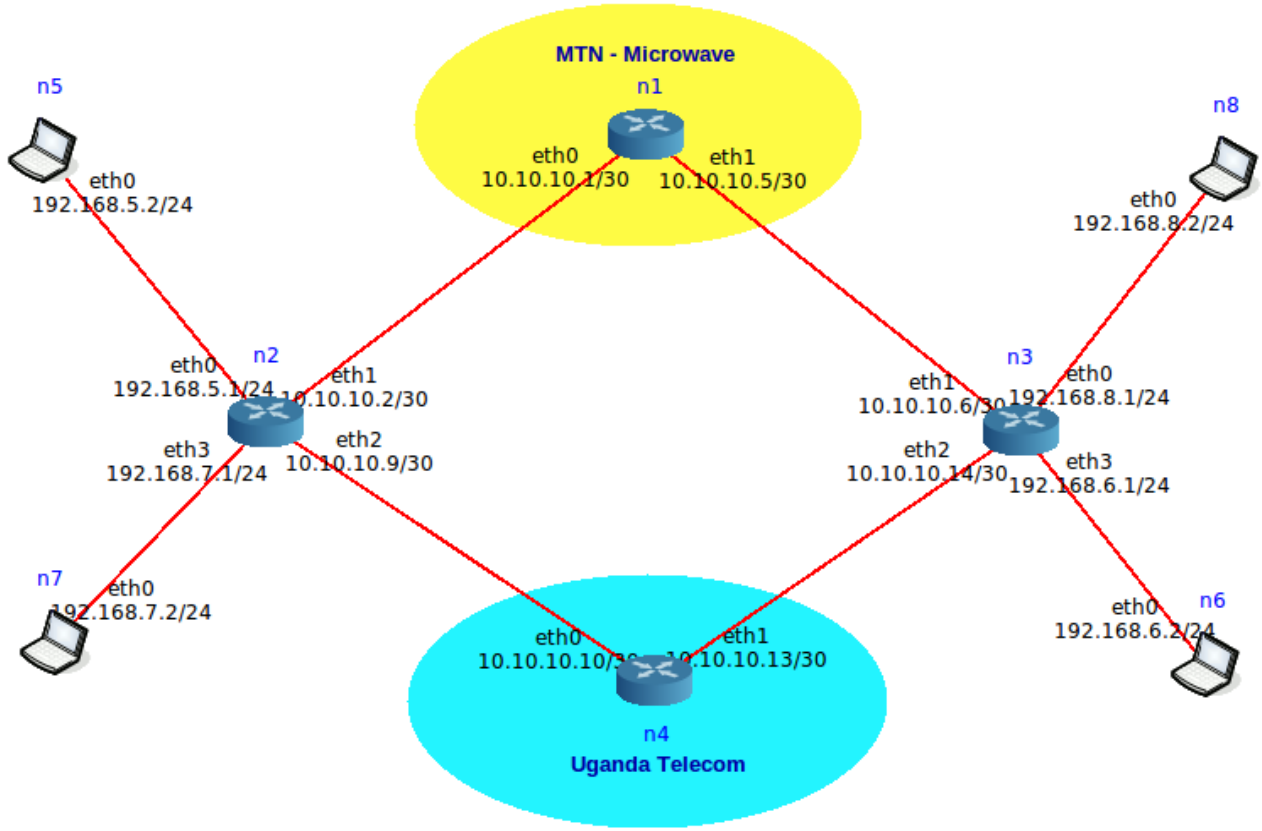
# Multipath, diversity management state machine

**Eng Diarmuid Ó Briain,**
Department of Electrical and Computer Engineering,
School of Engineering,
College of Engineering, Design, Art and Technology (CEDAT),
Makerere University
10 April 2016

This demonstrates the operation of a multipath and resilient link using **iproute2**.

# Network

The following is the network used. It was built in Network Training Emulator (NTE) v1.0.3.



# Create directory

```
nte@NTE-i386:~$ cd ~
nte@NTE-i386:~$ mkdir multipath
```

# Scripts and network

Copy the following attached files to **~/multipath** (**/home/nte/multipath**)

- Multipath-Routing.imn
- link_state.sh
- link_monitor.sh

**link_monitor.sh**
Operates on **Router n2 and Router n3** and monitors the link, it operates as a state machine switching between 4 possible states:

- State 0 : Both links are up
- State 1 : MTN up, UTL down
- State 2 : UTL up, MTN down
- State 3 : Both links are down

If the state changes **link_monitor.sh** will switch to the appropriate state.

**link_state.sh**

It is unrealistic to simply bring down an interface on the carrier network and back up again to demonstrate. This is because the link would come back up again without the routes that are internal to the carrier network. So I have created the **link_state.sh** script that will allow the simulation of the carrier coming up and down gracefully on either **Router n1** or **Router n4**.

## *Make the shell scripts executable.*

```
nte@NTE-i386:~$ sudo chmod +x ~/multipath/link_monitor.sh
nte@NTE-i386:~$ sudo chmod +x ~/multipath/link_state.sh
```

## *Copy the scripts to /usr/local/bin*

```
nte@NTE-i386:~$ sudo cp ~/multipath/link_*.sh /usr/local/bin
[sudo] password for nte: nte

nte@NTE-i386:~$ ls -la /usr/local/bin/link*
-rwxr-xr-x 1 root staff 1421 Apr 28 23:04 /usr/local/bin/link_monitor.sh
-rwxr-xr-x 1 root staff  733 Apr 28 23:04 /usr/local/bin/link_state.sh
```

# Run the network Multipath-Routing.imn

When ran in Common Open Research Emulator (CORE) in NTE, the network will start in a default state where both carriers, MTN and UTL are non-operational and without routes.

Note that there are no routes configured on either **Router n2** or **Router n3**, they come up with a routing table populated with local routes only.

```
/n2.conf# ip route list
10.10.10.0/30 dev eth1  proto kernel  scope link  src 10.10.10.2
10.10.10.8/30 dev eth2  proto kernel  scope link  src 10.10.10.9
192.168.5.0/24 dev eth0  proto kernel  scope link  src 192.168.5.1
192.168.7.0/24 dev eth3  proto kernel  scope link  src 192.168.7.1
```

Note: no routes to the networks 192.168.6.0/24, 192.168.8.0/24, 10.10.10.4/30, 10.10.10.12/30 or a default route.

```
n3.conf# ip route list
10.10.10.4/30 dev eth1  proto kernel  scope link  src 10.10.10.6
10.10.10.12/30 dev eth2  proto kernel  scope link  src 10.10.10.14
192.168.6.0/24 dev eth3  proto kernel  scope link  src 192.168.6.1
192.168.8.0/24 dev eth0  proto kernel  scope link  src 192.168.8.1
```

Note: no routes to the networks 192.168.5.0/24, 192.168.7.0/24, 10.10.10.0/30, 10.10.10.8/30 or a default route.

### *Ping test*

On **Host n5** or **n7** open a shell and ping **192.168.8.2** or **192.168.6.2**.

```
n5.conf# ping 192.168.8.2
PING 192.168.8.2 (192.168.8.2) 56(84) bytes of data.
From 192.168.5.1 icmp_seq=1 Destination Net Unreachable
From 192.168.5.1 icmp_seq=2 Destination Net Unreachable
```

Note that the ping fails for the lack of routes.

# Run the link monitor script

## *Starting point state 0, both carrier links are up*

Open shells on **Router n2** and **Router n3.** Now **r**un the **link_monitor.sh** script in each and see what happens.

Note: I am running it with **2> /dev/null** to offload the GNU/Linux standard error (stderr) messaging to /dev/null (the bit bucket) to keep the terminal clear for demonstration. It will work fine without this redirect, just that the shell is subjected to kernel messages over and above what is coming from the **link_monitor.sh**.

On **Router n2**:

```
n2.conf# link_monitor.sh 2> /dev/null

Router n2 dual link monitor
***************************
Diarmuid O'Briain

>>> Attempting to determine link state
>>> Trying to add routes to MTN and UTL, link is lost
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> Added multi-path route via both MTN and UTL
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
```

On **Router n3**:

```
n3.conf# link_monitor.sh 2> /dev/null

Router n3 dual link monitor
***************************
Diarmuid O'Briain

>>> Attempting to determine link state
>>> Trying to add routes to MTN and UTL, link is lost
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> Added multi-path route via both MTN and UTL
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
```

On **Host n5**:

```
In the n5 ping window the link comes up.

64 bytes from 192.168.8.2: icmp_seq=281 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=282 ttl=61 time=0.075 ms
64 bytes from 192.168.8.2: icmp_seq=283 ttl=61 time=0.061 ms
64 bytes from 192.168.8.2: icmp_seq=284 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=285 ttl=61 time=0.058 ms
64 bytes from 192.168.8.2: icmp_seq=286 ttl=61 time=0.059 ms
64 bytes from 192.168.8.2: icmp_seq=287 ttl=61 time=0.062 ms
```

### *Check the routing tables on Router n2 and Router n3*

Routes table in **Router n2**:

```
n2.conf# ip route list
default
       nexthop via 10.10.10.1  dev eth1 weight 1
       nexthop via 10.10.10.10  dev eth2 weight 2
10.10.10.0/30 dev eth1  proto kernel  scope link  src 10.10.10.2
10.10.10.4/30 via 10.10.10.1 dev eth1
10.10.10.8/30 dev eth2  proto kernel  scope link  src 10.10.10.9
10.10.10.12/30 via 10.10.10.10 dev eth2
192.168.5.0/24 dev eth0  proto kernel  scope link  src 192.168.5.1
192.168.7.0/24 dev eth3  proto kernel  scope link  src 192.168.7.1
```

Routes table in **Router n3**:

```
n3.conf# ip route list
default
       nexthop via 10.10.10.5  dev eth1 weight 1
       nexthop via 10.10.10.13  dev eth2 weight 2
10.10.10.0/30 via 10.10.10.5 dev eth1
10.10.10.4/30 dev eth1  proto kernel  scope link  src 10.10.10.6
10.10.10.8/30 via 10.10.10.13 dev eth2
10.10.10.12/30 dev eth2  proto kernel  scope link  src 10.10.10.14
192.168.6.0/24 dev eth3  proto kernel  scope link  src 192.168.6.1
192.168.8.0/24 dev eth0  proto kernel  scope link  src 192.168.8.1
```

### *Test multipath with iperf*

Enable the monitoring of throughput (Widgets → Throughput).

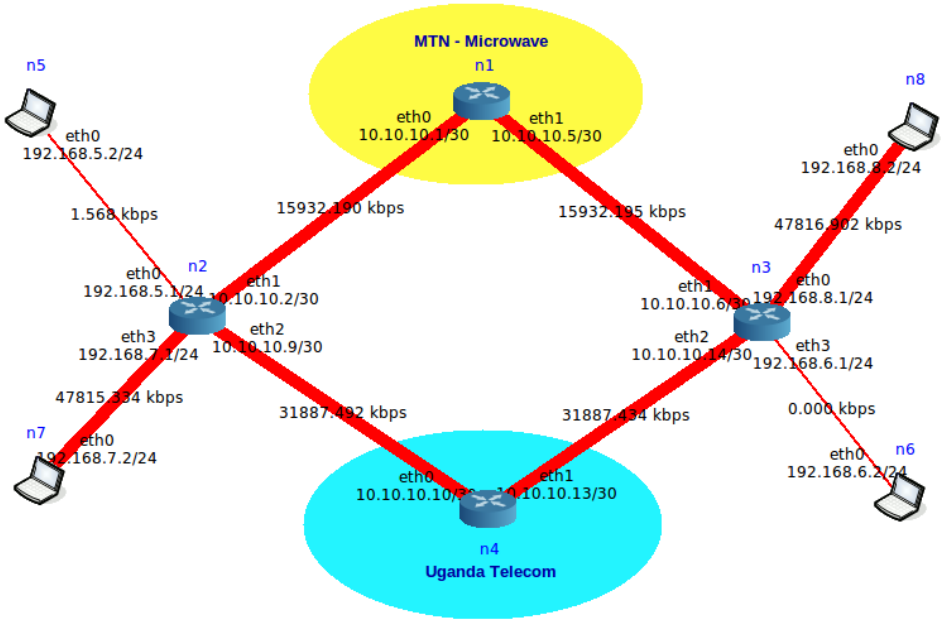
Launch an iperf server on **Host n8**:

```
n8.conf# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 192.168.8.2 port 5001 connected with 192.168.7.2 port 36381
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.0 sec  23.9 GBytes  20.6 Gbits/sec
```

Launch an iperf client on **Host n7**:

```
n7.conf# iperf -c 192.168.8.2
------------------------------------------------------------
Client connecting to 192.168.8.2, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 192.168.7.2 port 36382 connected with 192.168.8.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  23.0 GBytes  19.7 Gbits/sec
```

The diagram demonstrates that the traffic left **Host n7** and was sent over both the MTN and UTL paths to **Host n8**.



# link_state.sh script

## *Bring UTL down to change to state 1*

On the **Router n4** run the **link_state.sh** script to bring down the UTL carrier link. Note the state change on both **link_monitor.sh** shell windows.

```
n4.conf# link_state.sh down

Router n4 link state
Brings up or down eth1
**********************
Diarmuid O'Briain

Bringing Uganda Telecom Limited link down
```

On **Router n2**:

```
>>> Added bias route via MTN, diversity is lost
>>> State 1 : MTN up, UTL down, monitoring
>>> State 1 : MTN up, UTL down, monitoring
>>> State 1 : MTN up, UTL down, monitoring
```

On **Router n3**:

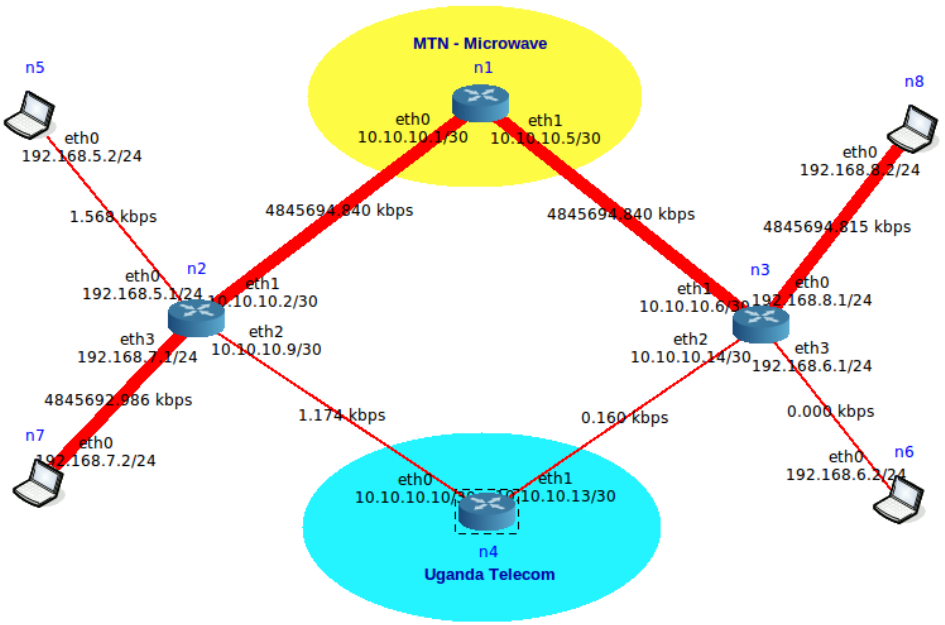
```
>>> Added bias route via MTN, diversity is lost
>>> State 1 : MTN up, UTL down, monitoring
>>> State 1 : MTN up, UTL down, monitoring
>>> State 1 : MTN up, UTL down, monitoring
```

On **Host n5** there are four packets lost as the UTL link comes down, then the service resumes as normal:

```
64 bytes from 192.168.8.2: icmp_seq=1097 ttl=61 time=0.067 ms
64 bytes from 192.168.8.2: icmp_seq=1098 ttl=61 time=0.061 ms
From 10.10.10.10 icmp_seq=1099 Destination Net Unreachable
64 bytes from 192.168.8.2: icmp_seq=1100 ttl=61 time=0.044 ms
From 10.10.10.10 icmp_seq=1101 Destination Net Unreachable
From 10.10.10.10 icmp_seq=1102 Destination Net Unreachable
From 10.10.10.10 icmp_seq=1104 Destination Net Unreachable
64 bytes from 192.168.8.2: icmp_seq=1105 ttl=61 time=0.061 ms
64 bytes from 192.168.8.2: icmp_seq=1106 ttl=61 time=0.055 ms
64 bytes from 192.168.8.2: icmp_seq=1107 ttl=61 time=0.094 ms
```

Running the **iperf** client on **Host n7** demonstrates the new path reality for the traffic is via MTN only.

```
n7.conf# iperf -c 192.168.8.2
------------------------------------------------------------
Client connecting to 192.168.8.2, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 192.168.7.2 port 36385 connected with 192.168.8.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  22.9 GBytes  19.6 Gbits/sec
```



### *Bring MTN down to change to state 3*

On the **Router n1** run the **link_state.sh** script to bring down the MTN carrier link also. Note the state change on both **link_monitor.sh** shell windows as both carrier links are out of order.

```
n1.conf# link_state.sh down

Router n1 link state
Brings up or down eth1
**********************
Diarmuid O'Briain

Bringing MTN Limited link down
```

On **Router n2** there is a state change to state 3:

```
>>> State 1 : MTN up, UTL down, monitoring
>>> Trying to add routes to MTN and UTL, link is lost
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> State 3 : UTL down, MTN down, panic, monitoring
```

On **Router n3** there is a state change to state 3 :

```
>>> State 1 : MTN up, UTL down, monitoring
>>> Trying to add routes to MTN and UTL, link is lost
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> State 3 : UTL down, MTN down, panic, monitoring
```

On Host **n5** there is catastrophic packet loss because both MTN and UTL are now offline:

```
64 bytes from 192.168.8.2: icmp_seq=1817 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=1818 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=1819 ttl=61 time=0.061 ms
64 bytes from 192.168.8.2: icmp_seq=1820 ttl=61 time=0.062 ms
From 10.10.10.1 icmp_seq=1821 Destination Net Unreachable
From 10.10.10.1 icmp_seq=1822 Destination Net Unreachable
From 10.10.10.1 icmp_seq=1823 Destination Net Unreachable
From 10.10.10.1 icmp_seq=1824 Destination Net Unreachable
From 192.168.5.1 icmp_seq=1837 Destination Net Unreachable
From 192.168.5.1 icmp_seq=1838 Destination Net Unreachable
From 192.168.5.1 icmp_seq=1839 Destination Net Unreachable
```

### *Bring MTN and UTL up to change to state 0*

On the **Router n1** and **Router n4** run the **link_state.sh** script to bring up the MTN and UTL carrier links up with the **up** option. Note the state change on both **link_monitor.sh** shell windows.

```
n1.conf# link_state.sh up

Router n1 link state
Brings up or down eth1
**********************
Diarmuid O'Briain

Bringing MTN Limited link up

n4.conf# link_state.sh up

Router n4 link state
Brings up or down eth1
**********************
Diarmuid O'Briain

Bringing Uganda Telecom Limited link up
```

On **Router n2** there is a state change to state 0:

```
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> Added multi-path route via both MTN and UTL
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
```

On **Router n3** there is a state change to state 0:

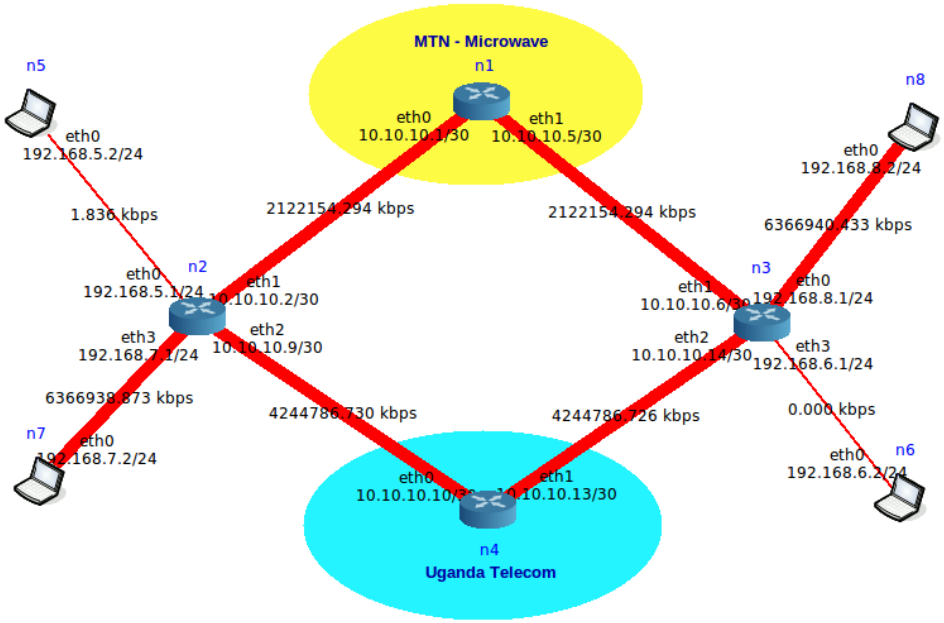
```
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> State 3 : UTL down, MTN down, panic, monitoring
>>> Added multi-path route via both MTN and UTL
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
```

On Host **n5** there is a throughput recovery as both MTN and UTL come online:

```
From 192.168.5.1 icmp_seq=1838 Destination Net Unreachable
From 192.168.5.1 icmp_seq=1839 Destination Net Unreachable
From 192.168.5.1 icmp_seq=1840 Destination Net Unreachable
64 bytes from 192.168.8.2: icmp_seq=2053 ttl=61 time=0.066 ms
64 bytes from 192.168.8.2: icmp_seq=2054 ttl=61 time=0.055 ms
64 bytes from 192.168.8.2: icmp_seq=2055 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2056 ttl=61 time=0.061 ms
64 bytes from 192.168.8.2: icmp_seq=2057 ttl=61 time=0.062 ms
```

Running the **iperf** client on **Host n7** demonstrates the new path reality for the traffic is back via both paths.

```
n7.conf# iperf -c 192.168.8.2
------------------------------------------------------------
Client connecting to 192.168.8.2, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 192.168.7.2 port 36387 connected with 192.168.8.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  22.0 GBytes  18.9 Gbits/sec
```



### *Bring MTN down to change to state 2*

On the **Router n1** run the **link_state.sh** script to bring down the MTN carrier link. Note the state change on both **link_monitor.sh** shell windows.

```
n1.conf# link_state.sh down

Router n1 link state
Brings up or down eth1
**********************
Diarmuid O'Briain

Bringing MTN Limited link down
```

On **Router n2** there is a state change to state 2:

```
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> Added bias route via UTL, diversity is lost
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
```

On **Router n3** there is a state change to state 2:

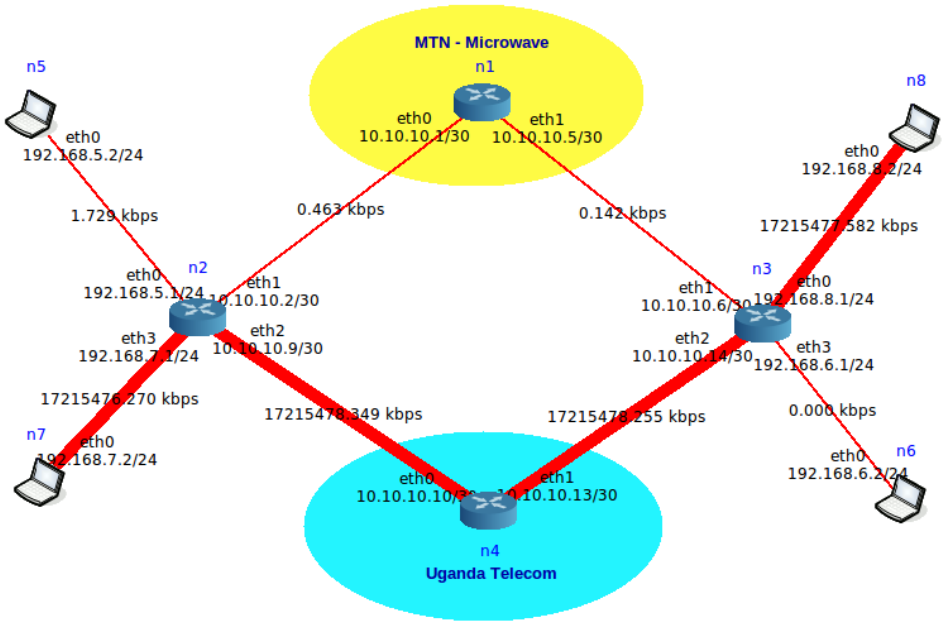
```
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> Added bias route via UTL, diversity is lost
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
```

On **Host n5** there is a failure of only one packet as the MTN link goes offline:

```
64 bytes from 192.168.8.2: icmp_seq=2335 ttl=61 time=0.065 ms
64 bytes from 192.168.8.2: icmp_seq=2336 ttl=61 time=0.057 ms
64 bytes from 192.168.8.2: icmp_seq=2337 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2338 ttl=61 time=0.057 ms
64 bytes from 192.168.8.2: icmp_seq=2339 ttl=61 time=0.059 ms
From 10.10.10.1 icmp_seq=2340 Destination Net Unreachable
64 bytes from 192.168.8.2: icmp_seq=2342 ttl=61 time=0.061 ms
64 bytes from 192.168.8.2: icmp_seq=2343 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2345 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2346 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2347 ttl=61 time=0.061 ms
```

Running the **iperf** client on **Host n7** demonstrates the new path reality via UTL on a single path.

```
n7.conf# iperf -c 192.168.8.2
------------------------------------------------------------
Client connecting to 192.168.8.2, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 192.168.7.2 port 36387 connected with 192.168.8.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  22.0 GBytes  18.9 Gbits/sec
```



### *Bring MTN up to change to state 0*

On the **Router n1** run the **link_state.sh** script to bring up the MTN carrier link. Note the state change on both **link_monitor.sh** shell windows is back to state 0, we have come full circle through all the states.

```
n1.conf# link_state.sh up

Router n1 link state
Brings up or down eth1
**********************
Diarmuid O'Briain

Bringing MTN Limited link up
```

On **Router n2** there is a state change to state 0:

```
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
>>> Added multi-path route via both MTN and UTL
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
```

On **Router n3** there is a state change to state 0:

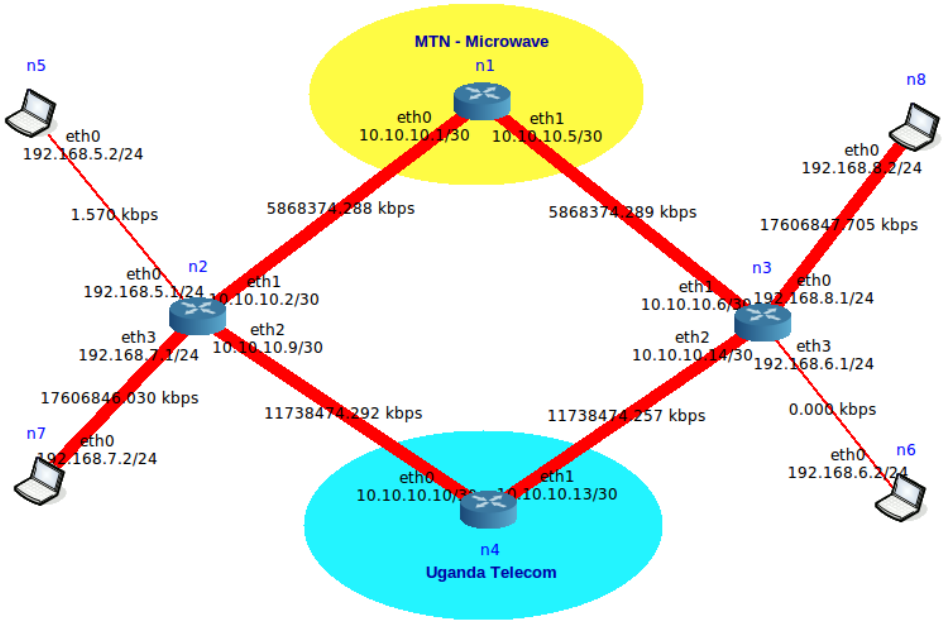
```
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
>>> State 2 : UTL up, MTN down, monitoring
>>> Added multi-path route via both MTN and UTL
>>> State 0 : Both links are up
>>> State 0 : Both links are up
>>> State 0 : Both links are up
```

On **Host n5** there is no packet loss through this state change where both links are active again:

```
64 bytes from 192.168.8.2: icmp_seq=2959 ttl=61 time=0.065 ms
64 bytes from 192.168.8.2: icmp_seq=2960 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2961 ttl=61 time=0.065 ms
64 bytes from 192.168.8.2: icmp_seq=2962 ttl=61 time=0.065 ms
64 bytes from 192.168.8.2: icmp_seq=2963 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2964 ttl=61 time=0.063 ms
64 bytes from 192.168.8.2: icmp_seq=2965 ttl=61 time=0.114 ms
64 bytes from 192.168.8.2: icmp_seq=2966 ttl=61 time=0.056 ms
64 bytes from 192.168.8.2: icmp_seq=2967 ttl=61 time=0.060 ms
64 bytes from 192.168.8.2: icmp_seq=2968 ttl=61 time=0.074 ms
64 bytes from 192.168.8.2: icmp_seq=2969 ttl=61 time=0.072 ms
64 bytes from 192.168.8.2: icmp_seq=2970 ttl=61 time=0.062 ms
```

Running the **iperf** client on **Host n7** demonstrates the new path reality for the traffic is back via both paths.

```
n7.conf# iperf -c 192.168.8.2
------------------------------------------------------------
Client connecting to 192.168.8.2, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 192.168.7.2 port 36389 connected with 192.168.8.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  20.5 GBytes  17.6 Gbits/sec
```



# Appendix 1: link_monitor.sh

```
nte@NTE-i386:~/multipath$ cat link_monitor.sh
#!/bin/bash

# Initialise variables

state=null
def_no=0

# Get the routers hostname
router_name=$(/bin/hostname)

echo
echo "Router $router_name dual link monitor"
echo '**************************'
echo "Diarmuid O'Briain"
echo

if [[ "$router_name" == 'n2' ]]; then
    MTN_IP='10.10.10.6'
    MTN_NextHop='10.10.10.1'
    MTN_NextNet='10.10.10.4/30'
    UTL_IP='10.10.10.14'
    UTL_NextHop='10.10.10.10'
    UTL_NextNet='10.10.10.12/30'
elif [[ "$router_name" == 'n3' ]]; then
    MTN_IP='10.10.10.2'
    MTN_NextHop='10.10.10.5'
    MTN_NextNet='10.10.10.0/30'
    UTL_IP='10.10.10.9'
    UTL_NextHop='10.10.10.13'
    UTL_NextNet='10.10.10.8/30'
else
    echo "This is $router_name and it is not one of the enterprise routers"
fi


while [ 1 ]; do

    sleep 5

    # Test links

    MTN_test=$(/usr/bin/fping -I  eth1 "$MTN_IP" | grep alive)
    UTL_test=$(/usr/bin/fping -I  eth2 "$UTL_IP" | grep alive)

    if [[ "$state" == 'null' ]]; then
        echo ">>> Attempting to determine link state"
    fi

    # // 4 possible states //
    # State 0 : Both links are up
    # State 1 : MTN up, UTL down
    # State 2 : UTL up, MTN down
    # State 3 : Both links are down

    # State 0 handler
    if [[ "$MTN_test" == *"alive"* && "$UTL_test" == *"alive"* ]]; then
        #echo "State 0 handler"
        if [[ "$state" == '0' ]]; then
            echo ">>> State 0 : Both links are up"
        else
            def_no=$(/sbin/ip route list | grep "default" | wc -l)
            until [[ "$def_no" == '0' ]]; do
                 /sbin/ip route del default
                 ((def_no--))
            done
            /sbin/ip route add default nexthop via "$MTN_NextHop" weight 1 nexthop via
"$UTL_NextHop" weight 2
```

```
            echo ">>> Added multi-path route via both MTN and UTL"
            echo ">>> State 0 : Both links are up"
            state=0
        fi

    # State 1 handler
    elif [[ "$MTN_test" == *"alive"* && "$UTL_test" != *"alive"* ]]; then
        if [[ "$state" == '1' ]]; then
            echo ">>> State 1 : MTN up, UTL down, monitoring"
        else
            def_no=$(/sbin/ip route list | grep "default" | wc -l)
            until [[ "$def_no" == '0' ]]; do
                 /sbin/ip route del default
                 ((def_no--))
            done
            /sbin/ip route add default via "$MTN_NextHop"
            /sbin/ip route add "$MTN_NextNet" via "$MTN_NextHop"
            /sbin/ip route add "$UTL_NextNet" via "$UTL_NextHop"
            echo ">>> Added bias route via MTN, diversity is lost"
            echo ">>> State 1 : MTN up, UTL down, monitoring"
            state=1
        fi

    # State 2 handler
    elif [[ "$MTN_test" != *"alive"* && "$UTL_test" == *"alive"* ]]; then
        if [[ "$state" == '2' ]]; then
            echo ">>> State 2 : UTL up, MTN down, monitoring"
        else
            def_no=$(/sbin/ip route list | grep "default" | wc -l)
            until [[ "$def_no" == '0' ]]; do
                 /sbin/ip route del default
                 ((def_no--))
            done
            /sbin/ip route add default via "$UTL_NextHop"
            /sbin/ip route add "$MTN_NextNet" via "$MTN_NextHop"
            /sbin/ip route add "$UTL_NextNet" via "$UTL_NextHop"
            echo ">>> Added bias route via UTL, diversity is lost"
            echo ">>> State 2 : UTL up, MTN down, monitoring"
            state=2
        fi

    # State 3 handler

    else
        if [[ "$state" == '3' ]]; then
            echo ">>> State 3 : UTL down, MTN down, panic, monitoring"
        else
            def_no=$(/sbin/ip route list | grep "default" | wc -l)
            until [[ "$def_no" == '0' ]]; do
                 /sbin/ip route del default
                 ((def_no--))
            done
            /sbin/ip route add "$MTN_NextNet" via "$MTN_NextHop"
            /sbin/ip route add "$UTL_NextNet" via "$UTL_NextHop"
            echo ">>> Trying to add routes to MTN and UTL, link is lost"
            echo ">>> State 3 : UTL down, MTN down, panic, monitoring"
            state=3
        fi

    fi

done

# End
```

# Appendix 2: link_state.sh

```
nte@NTE-i386:~/multipath$ cat link_state.sh
#!/bin/bash

# Get the routers hostname
router_name=$(/bin/hostname)

echo
echo "Router $router_name link state"
echo "Brings up or down eth1"
echo '**********************'
echo "Diarmuid O'Briain"
echo

if [[ "$router_name" == 'n1' ]]; then
    CARRIER='MTN Limited'
    DFT_IP='10.10.10.6'
elif [[ "$router_name" == 'n4' ]]; then
    CARRIER='Uganda Telecom Limited'
    DFT_IP='10.10.10.14'
else
    echo "This is $router_name and it is not one of the carrier routers"
fi

# Trap no entry on the command line

if [ $# -lt 1 ]; then
    echo "Usage:: link_state.sh <up | down> i.e. link_state.sh up"
    echo
    exit
fi

# Give options to user and execute for each
case $1 in
up)
  echo "Bringing $CARRIER link up"
  /sbin/ip link set dev eth1 up
  /sbin/ip route add 192.168.6.0/24 via "$DFT_IP" dev eth1
  /sbin/ip route add 192.168.8.0/24 via "$DFT_IP" dev eth1
  exit
  ;;
down)
  echo "Bringing $CARRIER link down"
  /sbin/ip link set dev eth1 down
  exit
  ;;
*)
  echo "Usage:: link_state.sh <up | down> i.e. link_state.sh up"
  echo
  exit
  ;;

esac

# End
```