Multipath TCP: Overview, Design, and Use-Cases

Benno Overeinder
NLnet Labs
... FOR MULTIPATH TCP

MPTCP slides by courtesy of Olivier Bonaventure (UCL)
The TCP Byte Stream Model

Client

ABCDEF...111232

Server

0988989 ... XYZZ

IP:1.2.3.4

IP:4.5.6.7

http://www.nlnetlabs.nl/
End Hosts Have Evolved

Mobile devices have multiple wireless interfaces

http://www.nlnetlabs.nl/
User Expectations
What Technology Provides

IP 1.2.3.4
What Technology Provides

3G cell tower

IP 1.2.3.4

IP 5.6.7.8

http://www.nlnetlabs.nl/
What Technology Provides

When IP addresses change TCP connections have to be re-established!
Data Centers
CHANGING INTERNET

http://www.nlnetlabs.nl/
The Internet Architecture That We Explain to Our Students

- Application
- Transport
- Network
- Datalink
- Physical

Switch 1

Hub 1

Router 1

Datalink
Physical

Network
Datalink
Physical
A Typical «Academic» Network

Application
Transport
Network
Datalink
Physical

Switch 1

Router 1

Application
Transport
Network
Datalink
Physical

http://www.nlnetlabs.nl/
In Reality

– almost as many middleboxes as routers
– various types of middleboxes are deployed

Figure 1: Box plot of middlebox deployments for small (fewer than 1k hosts), medium (1k-10k hosts), large (10k-100k hosts), and very large (more than 100k hosts) enterprise networks. Y-axis is in log scale.


http://www.nlnetlabs.nl/
A Middlebox Zoo

Web Security Appliance

VPN Concentrator

SSL Terminator

NAC Appliance

ACE XML Gateway

PIX Firewall Right and Left

Cisco IOS Firewall

IP Telephony Router

Streamer

Voice Gateway

Content Engine

NAT

http://www.nlnetlabs.nl/

http://www.cisco.com/web/about/ac50/ac47/2.html
How to Model Those Middleboxes?

- In the official architecture, they do not exist
- In reality...
TCP Segments Processed by a Router

<table>
<thead>
<tr>
<th>Ver</th>
<th>IHL</th>
<th>ToS</th>
<th>Total length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Identification</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flags Frag. Offset</td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Source IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source port</td>
<td>Destination port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acknowledgment number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>THL</td>
<td>Reserved</td>
<td>Flags</td>
<td>Window</td>
</tr>
<tr>
<td>Checksum</td>
<td>Urgent pointer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Payload

IP

TCP

Ver | IHL | ToS | Total length |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Identification</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flags Frag. Offset</td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Source IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source port</td>
<td>Destination port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acknowledgment number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>THL</td>
<td>Reserved</td>
<td>Flags</td>
<td>Window</td>
</tr>
<tr>
<td>Checksum</td>
<td>Urgent pointer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Payload

http://www.nlnetlabs.nl/
TCP Segments Processed by a NAT

<table>
<thead>
<tr>
<th>Ver</th>
<th>IHL</th>
<th>ToS</th>
<th>Total length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Frag. Offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Source IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source port</td>
<td>Destination port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acknowledgment number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>THL</td>
<td>Reserved</td>
<td>Flags</td>
<td>Window</td>
</tr>
<tr>
<td>Checksum</td>
<td>Urgent pointer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Payload

---

<table>
<thead>
<tr>
<th>Ver</th>
<th>IHL</th>
<th>ToS</th>
<th>Total length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Frag. Offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Source IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source port</td>
<td>Destination port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acknowledgment number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>THL</td>
<td>Reserved</td>
<td>Flags</td>
<td>Window</td>
</tr>
<tr>
<td>Checksum</td>
<td>Urgent pointer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Payload
TCP Segments Processed by an ALG Running on a NAT

<table>
<thead>
<tr>
<th>Ver</th>
<th>IHL</th>
<th>ToS</th>
<th>Total length</th>
<th>Identification</th>
<th>Flags</th>
<th>Frag. Offset</th>
<th>TTL</th>
<th>Protocol</th>
<th>Checksum</th>
<th>Source IP address</th>
<th>Destination IP address</th>
<th>Source port</th>
<th>Destination port</th>
<th>Sequence number</th>
<th>Acknowledgment number</th>
<th>THL</th>
<th>Reserved</th>
<th>Flags</th>
<th>Window</th>
<th>Checksum</th>
<th>Urgent pointer</th>
<th>Options</th>
<th>Payload</th>
</tr>
</thead>
</table>

| Ver | IHL | ToS | Total length | Identification | Flags | Frag. Offset | TTL | Protocol | Checksum | Source IP address | Destination IP address | Source port | Destination port | Sequence number | Acknowledgment number | THL | Reserved | Flags | Window | Checksum | Urgent pointer | Options | Payload |

http://www.nlnetlabs.nl/
Middleboxes don't change the Protocol field, but many discard packets with an unknown Protocol field.
MULTIPATH PROTOCOL
Design Objectives

• Multipath TCP is an *evolution* of TCP

• Design objectives
  – Support unmodified applications
  – Work over today’s networks
  – Works in all networks where regular TCP works
TCP Connection Establishment

• Three-way handshake

SYN, seq=1234, Options
SYN+ACK, ack=1235, seq=5678, Options
ACK, seq=1235, ack=5679
Data Transfer

seq=1234,"abcd"

ACK, ack=1238, win=4

seq=1238,"efgh"

ACK, ack=1242, win=0

http://www.nlnetlabs.nl/
Identification of a TCP Connection

Four tuple

- IP
- IP
- Port
- Port

All TCP segments contain the four tuple
The *New* Byte Stream Model
Design Decision

A Multipath TCP connection is composed of one or more regular TCP subflows that are combined.

- Each host maintains state that glues the TCP subflows that compose a Multipath TCP connection together.
- Each TCP subflow is sent over a single path and appears like a regular TCP connection along this path.
Multipath TCP and the Architecture


http://www.nlnetlabs.nl/
A regular TCP connection

• What is a regular TCP connection?
  – It starts with a three-way handshake
    • SYN segments may contain special options
  – All data segments are sent in sequence
    • There is no gap in the sequence numbers
  – It is terminated by using FIN or RST
Multipath TCP

SYN+Option

SYN+ACK+Option

ACK

SYN+OtherOption

SYN+ACK+OtherOption

ACK
How to Combine Two TCP Subflows?

How to link with blue subflow?
How to Link TCP Subflows?

SYN, Port_{src}=1234, Port_{dst}=80 + Option

SYN+ACK[...]

ACK

A NAT could change addresses and port numbers

SYN, Port_{src}=1235, Port_{dst}=80 + Option[link Port_{src}=1234, Port_{dst}=80]
How to Link TCP Subflows?

SYN, Port<sub>src</sub>=1234, Port<sub>dst</sub>=80
+Option[Token=5678]

SYN+ACK+Option[Token=6543]

ACK

MyToken=5678
YourToken=6543

SYN, Port<sub>src</sub>=1235, Port<sub>dst</sub>=80
+Option[Token=6543]

MyToken=6543
YourToken=5678
Subflow Agility

• Multipath TCP supports
  – addition of subflows
  – removal of subflows
How to Transfer Data?

seq=123,"a"

ack=124

seq=125,"c"

ack=126

seq=124,"b"

ack=125

seq=126,"d"

ack=127

http://www.nlnetlabs.nl/
How to Transfer Data in Today’s Internet?

seq=123,"a"

ack=124

seq=125,"c"

ack=126

Gap in sequence numbering space
Some DPI will not allow this!

seq=124,"b"

ack=125
Multipath TCP Data Transfer

• Two levels of sequence numbers
Multipath TCP Data Transfer

DSeq=0, seq=123, "a"

DAck=1, ack=124

DSeq=2, seq=124, "c"

DAck=3, ack=125

DSeq=1, seq=456, "b"

DAck=2, ack=457
TCP Congestion Control

• A linear rate adaption algorithm

- $rate(t + 1) = \alpha_C + \beta_C rate(t)$ when the network is congested
- $rate(t + 1) = \alpha_N + \beta_N rate(t)$ when the network is not congested

To be fair and efficient, a linear algorithm must use additive increase and multiplicative decrease (AIMD)

```python
# Additive Increase Multiplicative Decrease
if congestion:
    rate = rate * betaC  # multiplicative decrease, betaC<1
else:
    rate = rate + alphaN  # additive increase, v0>0
```
AIMD in TCP

• Congestion control mechanism
  – Each host maintains a congestion window \( (cwnd) \)
  – No congestion
    • Congestion avoidance (additive increase)
      – increase \( cwnd \) by one segment every round-trip-time
  – Congestion
    • TCP detects congestion by detecting losses
    • Mild congestion (fast retransmit – multiplicative decrease)
      – \( cwnd = cwnd/2 \) and restart congestion avoidance
    • Severe congestion (timeout)
      – \( cwnd = 1 \), set slow-start-threshold and restart slow-start
Evolution of the Congestion Window

- **Slow-start** exponential increase of cwnd
- **Congestion avoidance** linear increase of cwnd
- **Fast retransmit**
- Threshold
Congestion Control for Multipath TCP

- Simple approach
  - independant congestion windows
Independant Congestion Windows

- Problem

12Mbps
Coupling the Congestion Windows

• Principle
  – The TCP subflows are not independent and their congestion windows must be coupled

• EWTCP
  – For each ACK on path r, \( c\text{win}_r = c\text{win}_r + a/c\text{win}_r \) (in segments)
  – For each loss on path r, \( c\text{win}_r = c\text{win}_r / 2 \)

  – Each subflow gets window size proportional to \( a^2 \)
  – Same throughput as TCP if \( a = \frac{1}{\sqrt{n}} \)


http://www.nlnetlabs.nl/
Can We Split Traffic Equally Among All Subflows?

In this scenario, EWTCP would get 3.5 Mbps on the two hops path and 5 Mbps on the one hop path, less than the optimum of 12 Mbps for each Multipath TCP connection.

Linked Increases Congestion Control

• Algorithm
  – For each loss on path $r$, $c_{win_r} = c_{win_r}/2$
  
  – Additive increase

\[
c_{win_r} = c_{win_r} + \min \left( \max \left( \frac{cwnd_i}{(rtt_i)^2} \right), \frac{1}{cwnd_r} \right)
\]

Other Multipath-aware Congestion Control Schemes


USE-CASE: 3G AND WIFI
Usage of 3G and WiFi

• How should Multipath TCP use 3G and WiFi?

  – Full mode
    • Both wireless networks are used at the same time

  – Backup mode
    • Prefer WiFi when available, open subflows on 3G and use them as backup

  – Single path mode
    • Only one path is used at a time, WiFi preferred over 3G
Evaluation Scenario

WiFi:
Belgacom ADSL2+
(~8 Mbps, ~30 ms)

3G: Mobistar
(~2 Mbps, ~80 ms)
Recovery After Failure

Recovery After Failure

WiFi + 3G | 3G only

USE-CASE: SURFNET & CERN/CALTECH LHC NETWORK
Conclusion

• Multipath TCP is becoming a reality
  – Due to the middleboxes, the protocol is more complex than initially expected
  – RFC has been published
  – there is running code!
  – Multipath TCP works over today's Internet!

• What's next?
  – More use cases
    • IPv4/IPv6, anycast, load balancing, deployment
  – Measurements and improvements to the protocol
    • Time to revisit 20+ years of heuristics added to TCP
More Information

• Multipath TCP resources
  – http://nrg.cs.ucl.ac.uk/mptcp/

• Multipath TCP – Linux kernel implementation

• Multipath TCP – FreeBSD kernel implementation