FEASIBILITY STUDY ON MPTCP ACK VIA ALTERNATIVE PATH IN REAL NETWORK ENVIRONMENT

Hiroyuki KOIBUCHI, Hirotake ABE, Kazuhiko KATO
University of Tsukuba, Japan
Research Background

- Cloud computing technology is getting a lot of attention.
  - Cloud computing services such as AWS, Azure, and GCP.
  - Data centers are located in various regions of the world.

- Data centers will be used for providing various services including big data analyses.
  - Handle large amounts of data among data centers across countries.
  - Improving communication performance is very important to improve the overall performance or service quality.

- Network protocols affect communication performance.
  - MPTCP (Multi Path TCP) is a protocol that can improve communication performance.
MPTCP (Multi Path TCP)

• A TCP extension that allow two or more TCP connections to be used simultaneously
  • Applications can regard those as a single connection (w/ standard socket API)
• Acknowledgment message for a data packet is returned through the same path as the data packet go
  • This behavior can cause performance problem in some cases.

MPTCP extension to send ACK via alternative path

- **HayACK [CloudCom WS 2017]**
  - Sends ACK packets via alternative path.
  - Optimized to cases where there is a big difference in RTT (Round Trip Time) between two paths.
  - Increased throughput by up to 1.72x, average 1.52x.
    - Experiments on ns-3 simulations.
Possible discard of ACK packets caused by Middleboxes

- **Middlebox (MB):** General term for network appliances such as FW, NAT, VPN, TCP accelerator etc.
  - Most of those are dedicated for ensuring network security.
  - Discards “unusual” packets (e.g., inconsistent seq. no., unrelated to known TCP connections, etc.)
- **Practicality of ACK via alternative path can be greatly affected by the deployment of MBs.**
  - If it happens, extended MPTCP (like HayACK) needs to fall back to a single TCP connection.
Research Objective

• To survey how middleboxes are deployed in the current Internet that might interfere extended MPTCP that sends ACK via alternative path.
Related work (1):

"Is it still possible to extend TCP?"

• By Honda et al. [IMC' 11].
• They solicited volunteers from all over the world and asks to send “unusual” TCP packets that have inconsistent sequence number.
• Surveyed 142 routes from 24 countries
  • (jp:41 routes, uk:22 routes, us:17 routes, etc.)
• Only 67% of packets with inconsistent sequence numbers are arrived at the destination.

Related work(2):

Study on the availability of ECN function

• By Mirja et al. [PAM’ 13]
• They did a survey on compatibility of the Internet with ECN (Explicit Congestion Notification) packets
  • ECN introduces two additional flags to TCP packet format (ECE, CWR)
• Investigate the server’s responses to ECN packets.
  • Send packets with SYN, ECE, and CWR flags
  • 30% of 100,000 servers responded to ECN packets.

Our method

• Let client hosts in all over the world send “unusual” ACK packet to server and we count how many of those packets are successfully arrived at our server.
  • Using a dedicated program made with Scapy, an API to tweak TCP/IP packets from a program in Python.
• Survey #1: (simpler)
  • Investigation with “spontaneous” ACK, which means the ACK that is not related to any known connections.
• Survey #2: (more complex)
  • Investigation with ACK packets that have inconsistent sequence number under “mimicked” TCP connections.
Our challenge: how to collect as many client hosts as possible in a very limited time?

Our solution: Use VPN service!

VPNGate is a free-of-charge VPN service hosted by many (approx. 15k) volunteers all over the world.

Breaking a government censorship is their main objective.
How it works with VPNGate

- Our client host picks one of the VPNGate servers and establishes a VPN connection to the selected server.
  - They provide the latest list of available servers on their website (vpngate.net)
- Our client host sends various packet to our server via the VPN connection.
- Our server counts how many packets sent from our client host are successfully arrived.
How survey #1 ("spontaneous") works

• Send survey packets from various IP addresses without connections.
  • If a packet is sent via VPN, its sender address become VPN server’s, not client host’s.
How survey #2 (“mimicked”) works

- Send survey packets from various IP addresses after establishing connections using mimicked 3-way handshake with Scapy.
Packet types used in the surveys

• Survey #1: “spontaneous” (simpler)
  • Ports: 80(http), 443(https), 34343
  • Flags: Syn, Ack, Syn+Ack, Syn+Ece, Ack+Ece

• Survey #2: “mimicked” (more complex)
  • Port: 80 (http)
  • Flags: Ack, Syn+Ack, Ack+Ece, Ack+Cwr, Ack+Cwr+Ece

• We used ECN flags because we thought it might affect the results.
A bug we found in VPNGate

- VPNGate uses SoftetherVPN as its building block.
- SoftetherVPN (v4.31 or earlier) had a bug in handling TCP packets with ECN flags.
  - It discards every TCP packets with both Syn and ECN flags!
  - Ironically, older SoftetherVPN might be a MB that prevents deploying new protocols... 😞
- We reported the bug and they fixed it in v4.32.
- We had to discard some of results related to ECN flags but others were unaffected.
Results of survey #1 “spontaneous” (including affected results)

• Only 11% of VPNGate nodes were fixed at the time we conducted the survey.

<table>
<thead>
<tr>
<th>country</th>
<th>IP</th>
<th>S:80</th>
<th>S:443</th>
<th>S:34343</th>
<th>SE all</th>
<th>SA all</th>
<th>AE all</th>
<th>A all</th>
</tr>
</thead>
<tbody>
<tr>
<td>jp</td>
<td>73</td>
<td>73</td>
<td>73</td>
<td>73</td>
<td>14</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>kr</td>
<td>76</td>
<td>76</td>
<td>76</td>
<td>76</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>us</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>hk</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>id</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ve</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vn</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>my</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ph</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ro</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>th</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SUM</td>
<td>163</td>
<td>163</td>
<td>163</td>
<td>163</td>
<td>18</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>%</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>11</td>
<td>0</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

S:Syn, A:Ack, E:Ece
Results of survey #1 “spontaneous” (w/o affected results)

- n=18. (was 163 before filtering)
- We are still able to know few things even from these results.

- 67% of “spontaneous” Ack was not filtered.
- Adding ECN flags to Ack does not affect at all.

<table>
<thead>
<tr>
<th>country</th>
<th>IP</th>
<th>S:80</th>
<th>S:443</th>
<th>S:34343</th>
<th>SE all</th>
<th>SA all</th>
<th>AE all</th>
<th>A all</th>
</tr>
</thead>
<tbody>
<tr>
<td>jp</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>kr</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>us</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>hk</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>id</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ve</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vn</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>my</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ph</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ro</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>th</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SUM</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>18</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>%</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>0</td>
<td>67</td>
<td>67</td>
</tr>
</tbody>
</table>
Results of survey #2 “mimicked”

- Not affected by the buggy VPNGate because Syn was already sent without ECN flags by mimicked 3-way handshake

- Mimicking can improve acceptance of ACK with inconsistent seq. no. (67% => 82%)
- Again, adding ECN flags to Ack does not affect at all.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>KR</td>
<td>44</td>
<td>43</td>
<td>43</td>
<td>0</td>
<td>43</td>
<td>43</td>
</tr>
<tr>
<td>JP</td>
<td>31</td>
<td>21</td>
<td>21</td>
<td>0</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>VN</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>RS</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>AR</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>IN</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CL</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>NO</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TH</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>KH</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>LV</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>合計</td>
<td>87</td>
<td>71</td>
<td>71</td>
<td>0</td>
<td>71</td>
<td>71</td>
</tr>
<tr>
<td>%</td>
<td>100</td>
<td>82</td>
<td>82</td>
<td>0</td>
<td>82</td>
<td>82</td>
</tr>
</tbody>
</table>
Summary of the results

- Acceptance rate of “unusual” ACK packets looks increasing in last decade.
  - In 2011: 67% (by Honda et al.)
  - In 2021: 82% (this work)
- Mimicking 3-way handshake can increase the acceptance rate.
  - 67% (w/o handshake) => 82% (w/ handshake)
- ECN flags does not affect the drop rate.
Conclusion

- We conducted surveys on acceptance rate of TCP ACK packets that have inconsistent sequence numbers.
  - It happens when we use extended MPTCP like HayACK.
- Our results show a sign that the situation of interference caused by middleboxes is getting better.
  - Broader chance that newer protocols like HayACK can work.
ACKnowledgment (that I should not drop)

• This work was supported by JSPS KAKENHI Grant Number 18K11235.

That’s all. Thank you for listening!