Is it still possible to extend TCP?

Authors: Michio Honda, Keio University, Fujisawa, Japan
Yoshifumi Nishida, Keio University, Fujisawa, Japan
Costin Raicu, Universitatea Politehnica Bucuresti, Romania
Adam Greenhlgh, University College London, United Kingdom
Mark Handley, University College London, United Kingdom
Hideyuki Tokuda, Keio University, Fujisawa, Japan

Presenter: Alexandros Tsalidis
Technische Universität München
Munich, 1. June 2017

Content

Introduction

- Internet was designed to be extensible but things changed
- TCP options to extend functionality
- Invasion of middleboxes
  - Interaction with TCP options?

Contribution

✓ Effects of middleboxes on TCP options

✓ What middleboxes are doing to traffic

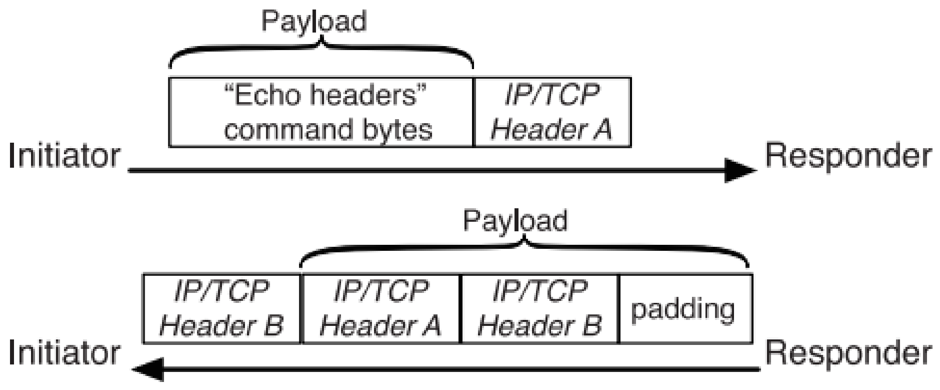
✓ Implications for protocol designers

Methodology and Datasets

Testing tool: TCPExposure

- Client and Server tool

Parameter	Initiator	Responder
Initial Sequence Num (ISN)	252001	11259375
Window Size	8064	32768
MSS	512	512
Window Scale	-	6
SACKOK	-	1
Timestamp (TS_val)	-	12345678



Methodology and Datasets

Port 80 (http)

Port 443 (https)

Port 34343 (unassigned)

Measurements target access networks, where ISPs deploy middleboxes to improve the experience of the customers

Six types of access networks:

- Home
- Hotspot
- Cellular
- University
- Enterprise
- Hosting

TCP Options test

The following tests were performed:

1. **Unknown option in SYN.** The SYN and SYN/ACK segments include an unregistered option.
2. **Unknown option in Data segment.** The test includes unknown options in data segments sent by client and server.
3. **Known option in Data segment.** The test includes a well-known option in data segments sent by client and server.

We tested for:

- SYN is passed unmodified.
- SYN containing the option is dropped.
- SYN is received, but option was removed.
- Connection is reset by the middlebox.

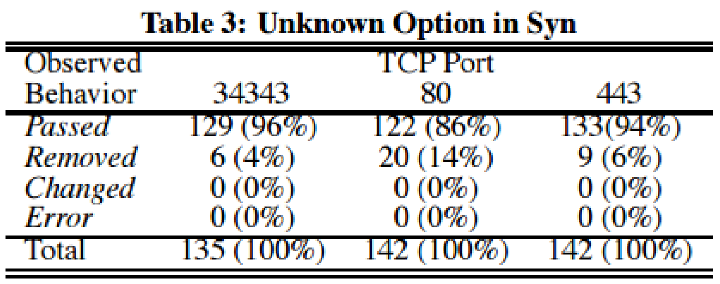


Table 3: Unknown Option in Syn

Observed	TCP Port		
Behavior	34343	80	443
Passed	129 (96%)	122 (86%)	133(94%)
Removed	6 (4%)	20 (14%)	9 (6%)
Changed	0 (0%)	0 (0%)	0 (0%)
Error	0 (0%)	0 (0%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

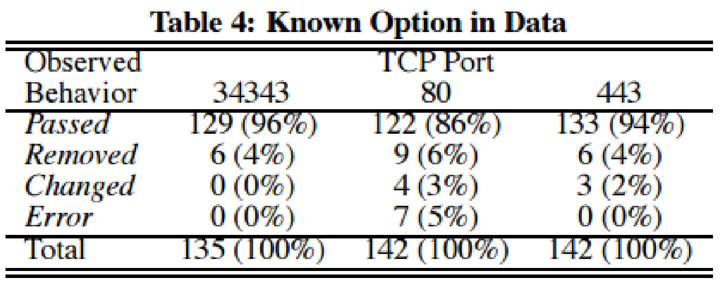


Table 4: Known Option in Data

Observed	TCP Port		
Behavior	34343	80	443
Passed	129 (96%)	122 (86%)	133 (94%)
Removed	6 (4%)	9 (6%)	6 (4%)
Changed	0 (0%)	4 (3%)	3 (2%)
Error	0 (0%)	7 (5%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

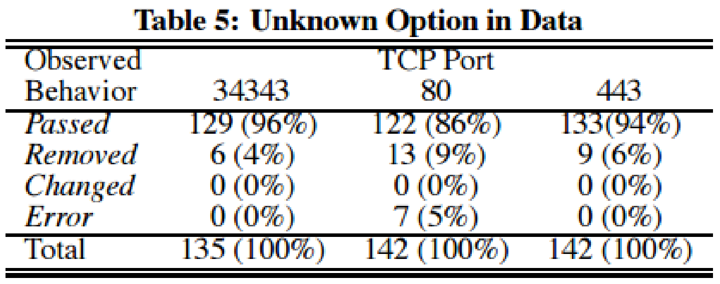
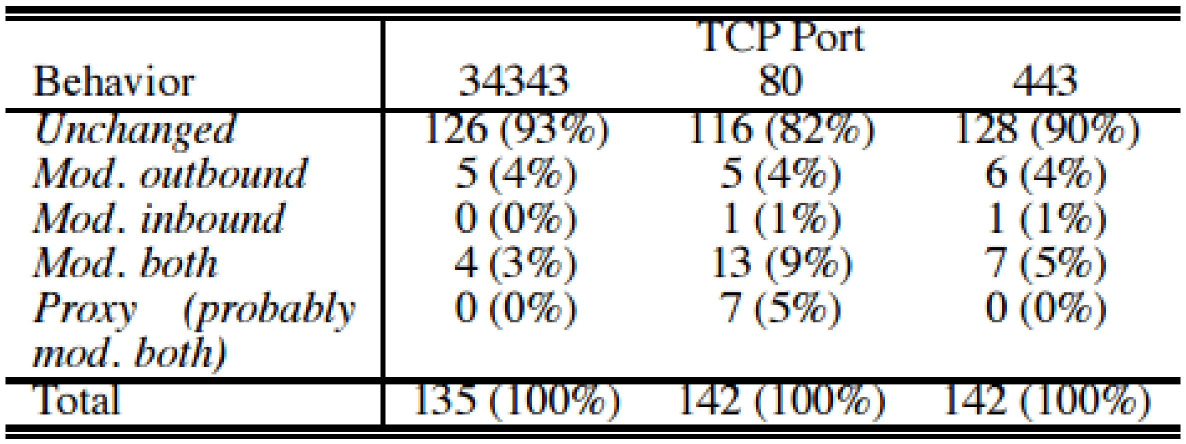


Table 5: Unknown Option in Data

Observed	TCP Port		
Behavior	34343	80	443
Passed	129 (96%)	122 (86%)	133(94%)
Removed	6 (4%)	13 (9%)	9 (6%)
Changed	0 (0%)	0 (0%)	0 (0%)
Error	0 (0%)	7 (5%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

Sequence number modification

We examine whether middleboxes modify the sequence numbers sent by the end systems



	TCP Port		
Behavior	34343	80	443
Unchanged	126 (93%)	116 (82%)	128 (90%)
Mod. outbound	5 (4%)	5 (4%)	6 (4%)
Mod. inbound	0 (0%)	1 (1%)	1 (1%)
Mod. both	4 (3%)	13 (9%)	7 (5%)
Proxy (probably mod. both)	0 (0%)	7 (5%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

Finally it is unsafe to embed sequence numbers in TCP options!

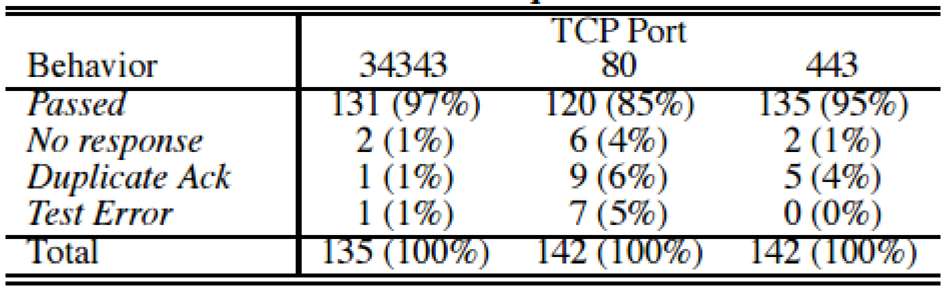
Sequence space holes

TCP is a reliable protocol. Its cumulative ACK does not move forward unless all segments are received

Two ways for a middlebox to observe if that does not happen:

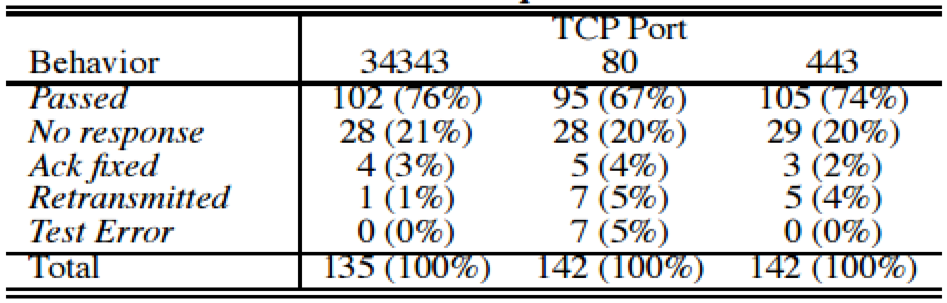
- Data-First
- Ack-First

Table 10: Data-First Sequence Hole Test



	TCP Port		
Behavior	34343	80	443
Passed	131 (97%)	120 (85%)	135 (95%)
No response	2 (1%)	6 (4%)	2 (1%)
Duplicate Ack	1 (1%)	9 (6%)	5 (4%)
Test Error	1 (1%)	7 (5%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

Table 11: Ack-first Sequence Hole Test



	TCP Port		
Behavior	34343	80	443
Passed	102 (76%)	95 (67%)	105 (74%)
No response	28 (21%)	28 (20%)	29 (20%)
Ack fixed	4 (3%)	5 (4%)	3 (2%)
Retransmitted	1 (1%)	7 (5%)	5 (4%)
Test Error	0 (0%)	7 (5%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

The conclusion of this test was that TCP extensions relying on sequence number holes are unsafe!

Inconsistent Retransmission

What happens if a TCP sender retransmits a packet, but includes different data than the original in the retransmission?

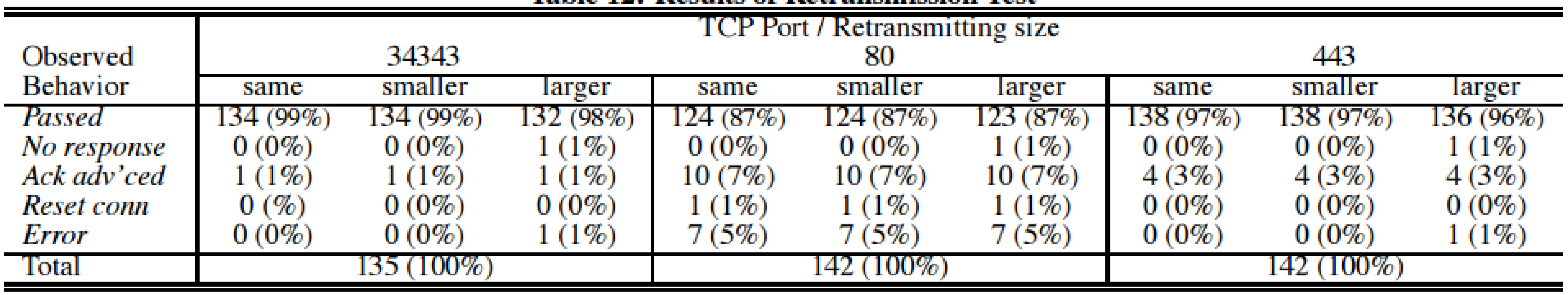


Table 12: Results of Retransmission Test

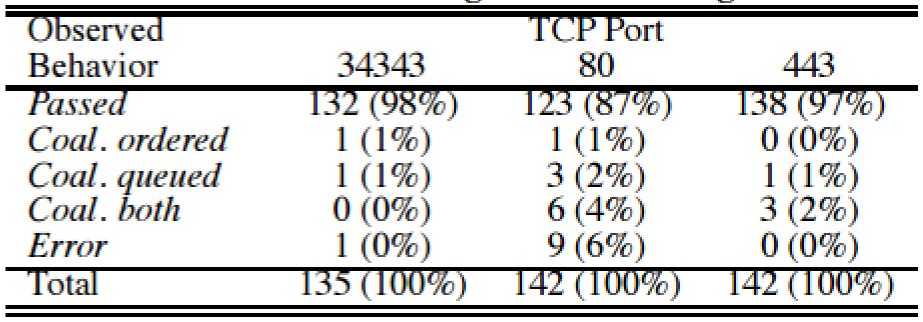
Observed	TCP Port / Retransmitting size								
	34343			80			443		
Behavior	same	smaller	larger	same	smaller	larger	same	smaller	larger
Passed	134 (99%)	134 (99%)	132 (98%)	124 (87%)	124 (87%)	123 (87%)	138 (97%)	138 (97%)	136 (96%)
No response	0 (0%)	0 (0%)	1 (1%)	0 (0%)	0 (0%)	1 (1%)	0 (0%)	0 (0%)	1 (1%)
Ack adv'ced	1 (1%)	1 (1%)	1 (1%)	10 (7%)	10 (7%)	10 (7%)	4 (3%)	4 (3%)	4 (3%)
Reset conn	0 (%)	0 (0%)	0 (0%)	1 (1%)	1 (1%)	1 (1%)	0 (0%)	0 (0%)	0 (0%)
Error	0 (0%)	0 (0%)	1 (1%)	7 (5%)	7 (5%)	7 (5%)	0 (0%)	0 (0%)	1 (1%)
Total	135 (100%)			142 (100%)			142 (100%)		

Any extension that wished to use inconsistent retransmissions would encounter few problems, so long as it did not matter greatly whether the original or the retransmission actually arrives.

Re-segmentation

- To test segment splitting, we simply send a full-sized segment
- The opposite of segment splitting is segment coalescing, where a middlebox combines two or more segments into a larger segment

Table 13: Results of Segment Coalescing Test



Observed	TCP Port		
Behavior	34343	80	443
Passed	132 (98%)	123 (87%)	138 (97%)
Coal. ordered	1 (1%)	1 (1%)	0 (0%)
Coal. queued	1 (1%)	3 (2%)	1 (1%)
Coal. both	0 (0%)	6 (4%)	3 (2%)
Error	1 (0%)	9 (6%)	0 (0%)
Total	135 (100%)	142 (100%)	142 (100%)

Although middleboxes do split and coalesce segments, none did so while passing unknown options. So it seems relatively safe to assume that if an option is passed, it arrives with the segment on which it was sent

Intelligent NICs

- Even Network Interface Cars can have embedded TCP knowledge
- We are concerned in particular with TCP Segmentation Offload (TSO), where the host OS sends large segments and relies on the NIC to resegment to match the MTU or the receiver’s MSS
- 12 NICs were tested and they all correctly copied the options to all the split segments
- The implication is that TCP options must be designed so that when they are duplicated on consecutive segments, this does not adversely affect correctness or performance

Implications for protocol designers

- Multipath TCP
  Decisions were dictated by middleboxes
- TcpCrypt
  Tightly constrained by middleboxes for its solutions
- Extending TCP option space
  MPTCP and TcpCrypt use relatively large option space and leave very little free. So Long Option can be deployed with some concerns

Conclusion

- It is still possible to extend TCP using the intended mechanism – TCP options although there are some guidelines and assumptions we should not take for granted
- Based on that information the three TCP extensions we checked had made sensible choices and are tightly constrained by middlebox behaviors to their chosen solutions
- We urge middlebox designers to consider explicitly whether they want to allow new TCP extensions when implementing certain functionality

Thank you for your attention!