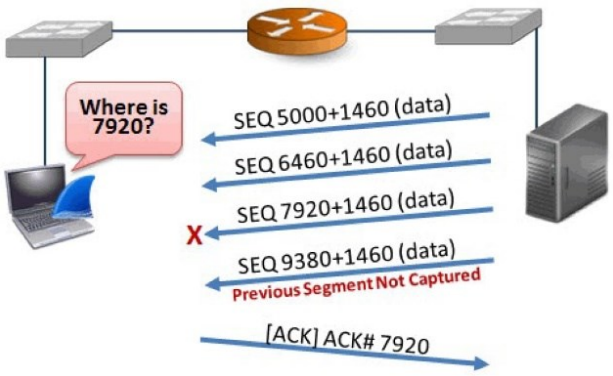
กิจกรรมที่ 7 : TCP Retransmission

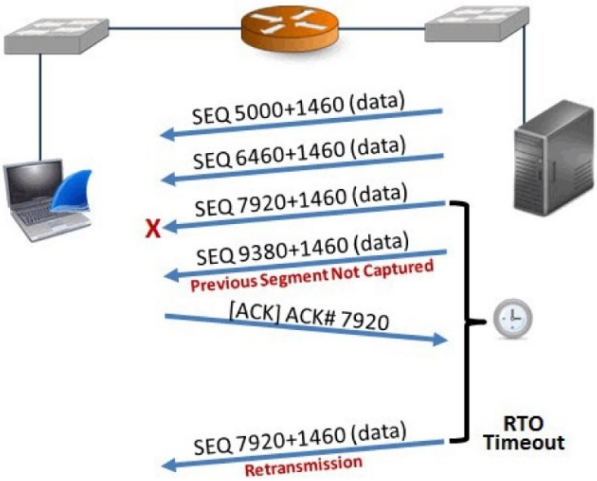
กิจกรรมครั้งนี้จะเป็นการทำความเข้าใจกับโปรโตคอล TCP (Transmission Control Protocol) ให้มากยิ่งขึ้น โดยเน้นเรื่องของกระบวนการ Retransmission

การรับข้อมูลของ TCP จะมีแนวทางการตอบ ACK ที่ระบุ ACK# เป็นหมายเลข X เพื่อใช้บ่งบอกว่าได้รับข้อมูลที่มี SEQ# ก่อน X ทั้งหมดแล้ว และกำลังรอรับ SEQ# X เป็นตัวถัดไป (Cumulative ACK) โดยทั่วไปสามารถสรุปแนวทางได้ดังตารางข้างล่างนี้

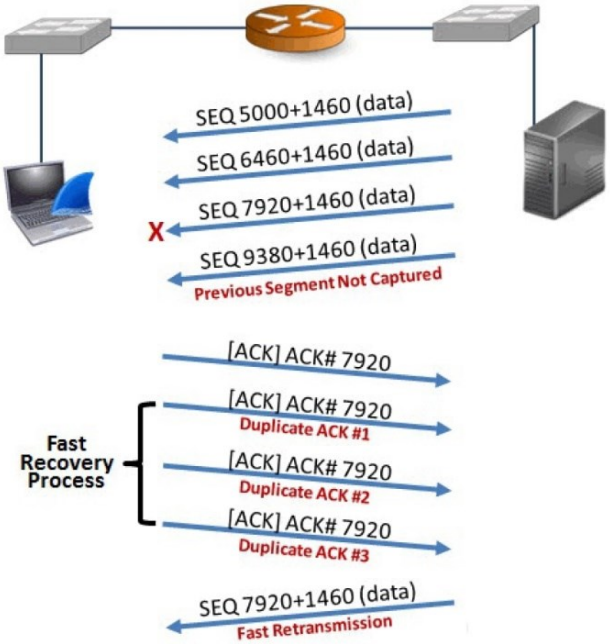
ข้อมูลที่ฝั่งรับส่ง ACK ตอบไปแล้ว	TCP Segment ที่พึ่งได้รับมา	สิ่งที่ฝั่งรับต้องดำเนินการต่อไป
ตอบ ACK# เท่ากับ SEQ# ที่กำลังรอรับ	SEQ# ตรงกับที่กำลังรอรับ	ชะลอการส่ง ACK (Delayed ACK) ออกไปก่อน แต่ไม่เกิน 500 ms (ตาม RFC 5681) โดยหากไม่ได้รับ TCP Segment อื่นเพิ่มในเวลาที่กำหนดจึงส่ง ACK ออกไป
ยังตอบ ACK# ไม่ถึง SEQ# ที่กำลังรอรับ	SEQ# ตรงกับที่กำลังรอรับ	ส่ง ACK ออกไปทันที โดยระบุ ACK# เป็น SEQ# ตัวที่กำลังรอรับล่าสุด ซึ่งถัดไปข้อมูลใน TCP Segment ที่พึ่งได้รับมา
ตอบ ACK# จนถึง SEQ# ใดๆ	SEQ# เกินกว่าที่กำลังรอรับ ข้อมูลไม่เป็นไปตามลำดับ (Out of Order)	ส่ง ACK ออกไปทันที โดยระบุ ACK# เป็น SEQ# ตัวที่กำลังรอรับอยู่ ซึ่งอาจจะมี ACK# ซ้ำกับ ACK ล่าสุดที่เคยส่งออกไปแล้ว (Duplicate ACK)



- ในกรณีที่เกิด Lost Segment จะมีวิธีการแก้ไข 2 รูปแบบ คือ Retransmission โดยฝั่งส่งทำการส่งข้อมูลใหม่เมื่อฝั่งส่งไม่ได้รับ ACK ภายในเวลา Retransmission Timeout (RTO)



- อีกรูปแบบหนึ่ง คือ Fast Retransmission ซึ่งจะใช้ได้เฉพาะ OS ที่สนับสนุน โดยเมื่อฝั่งส่งได้รับ Duplicate ACK ครบ 3 ครั้ง ก็จะส่งข้อมูลให้ฝั่งรับใหม่



1. ให้เปิดไฟล์ http-browse101d.pcapng คลิกขวาที่ Sequence Number และเลือก Apply as Column และตั้งชื่อว่า SEQ# จากนั้นคลิกขวาที่ Next Sequence Number และเลือก Apply as Column และตั้งชื่อว่า NEXTSEQ# และคลิกขวาที่ Acknowledgment Number และเลือก Apply as Column และตั้งชื่อว่า ACK# จัดรูปแบบคอลัมน์ให้เหมาะสม จะเห็นว่าเรามีข้อมูลของ SEQ#, NEXTSEQ# และ ACK# สำหรับช่วยในการวิเคราะห์
2. ใน Wireshark จะมีข้อมูลที่ Wireshark วิเคราะห์ขึ้น และสามารถนำมาเป็น Display Filter ได้ เช่น
    - tcp.analysis.duplicate_ack จะค้นหา Packet ที่เป็น Duplicate ACK
    - tcp.analysis.lost_segment จะค้นหากรณีเกิด Lost Segment

- tcp.analysis.retransmission จะค้นหา Packet ที่เกิดจากการทำ Retransmission
- tcp.analysis.fast_retransmission จะค้นหา Packet ที่เกิดจากการทำ Fast Retransmission

3. ให้เปิดไฟล์ tr-general101d.pcapng แล้วใช้ tcp.analysis.lost_segment กรอง จะพบว่ามี Lost Segment ทั้งหมด 5 แห่ง จาก Packet 10417 ให้ย้อนดู Packet 10416 แล้วตอบคำถามว่า มีข้อมูลหายไปเท่าไร มี Packet หายไปกี่ Packet บอกวิธีการหาแบบย่อๆ

   ก่อนอื่นให้มองภาพรวมก่อน เหตุเริ่มแรก คือ ข้อมูลหายระหว่าง Packet 10416-10417 โดย 10416 มี SEQ# เป็น 9163441 และมี Next SEQ# เป็น 9164761 แต่ใน 10417 กลับมี SEQ# เป็น 9175321 ซึ่งเมื่อนำมาลบกัน (9175321 – 9164761) เท่ากับ 10560 ไบต์ และเมื่อดู Segment Len ซึ่งมีค่าเท่ากับ 1320 เมื่อนำ 10560/1320 = 8 ดังนั้นจำนวน Packet ที่หายไป คือ 8 Packet หรือ 8 Segment

Segment	SEQ#	NEXT SEQ#	ACK#	
10416	9163441	9164761		
10417	9175321	9176641		Previous segment not captured

4. จาก Lost Segment ใน Packet 10417 หลังจากนั้นจะพบว่ามี Duplicate Ack เกิดขึ้นเป็นจำนวนมาก ให้อธิบายสาเหตุของการเกิด Duplicate ACK และเกิด Duplicate ACK กี่ครั้งในกรณีนี้

   ต่อมา เมื่อฝั่งรับไม่ได้รับข้อมูลที่หายไป แต่ได้รับ Packet 10417 จนถึง Packet 12033 ซึ่งมี SEQ# ไม่ตรงกับ SEQ# ที่ฝั่งรับคาดหวัง คือ 9164761 จึงส่ง Acknowledge Packet ที่มี ACK# = 9164761 กลับมา เพื่อแจ้งฝั่งส่งว่า Packet ที่มี SEQ# = 9164761 ยังไม่ได้รับ ซึ่งเนื่องจากมีค่า ACK# ซ้ำกับที่เคยส่งมาก่อนหน้านี้ จึงถือว่าเป็น Duplicate Ack โดยมีจำนวน Duplicate Ack = ((12033-10417)/2) = 808 ครั้ง
   หรือสามารถดูได้จาก Analyze -> Expert Information ภายใต้หัวข้อ Duplicate ACK จะเห็นว่า มี Duplicate Ack ที่ซ้ำกับ 10418 ตัวสุดท้ายคือ [TCP Dup ACK 10418#808] จึงสรุปได้ว่ามีซ้ำ 808 ครั้ง

   อย่างไรก็ตามในระหว่าง Packet 10417 และ 12033 ยังมี Lost Segment เกิดขึ้นอีก 1 แห่งคือที่ Packet 11497 และ 11499 ซึ่งข้อมูลหายไป = 9901321 – 9889441 = 11880 ไบต์ คิดเป็น 9 Packet

Packet	SEQ#	NEXT SEQ#	ACK#	
10416	9163441	9164761		
10417	9175321	9176641		Previous segment not captured 10560 ไบต์
10418	1	1	9164761	ACK# = 9164761 ครั้งที่ 1
10420-11497	9176641	9889441	9164761	Dup ACK [#1 - #540]
11499	9901321	9902641		Previous segment not captured 11880 ไบต์
11501-12034	9902641	10255081	9164761	Dup ACK [#541 - #808]

จากข้อ 3 ข้อมูลที่หายไป ผู้ส่งทราบเมื่อใด ได้มีการส่งใหม่หรือไม่ และส่งใหม่ใน Packet ใด และเวลาผ่านไปนานเท่าใดถึงได้ส่งใหม่

ต่อมาใน Packet 12035 มีการส่งข้อมูลที่มี SEQ# = 9164761 ซึ่งเป็นข้อมูลที่ตรงกับลำดับที่ฝั่งรับคาดหวัง จึงถือเป็นการส่งใหม่สำหรับ 9164761 จึงถือว่ามีการส่งใหม่สำหรับข้อมูลลำดับ 9164761 แล้ว สำหรับเวลาที่ใช้เมื่อนำเวลาของ Packet 12035 – เวลาของ Packet 10417 จะได้เท่ากับ 0.465989 วินาที (ถ้านับจาก 10416 จะได้ 0.476811 วินาที) แต่ถ้านับจาก Duplicate ACK สุดท้ายจะได้เท่ากับ 0.000033 วินาที

ต่อมาใน Packet 12036 ก็มีการส่งข้อมูลลำดับ 10255081 ไป เนื่องจากฝั่งส่งไม่ทราบว่าฝั่งรับได้รับข้อมูลใดบ้าง ทราบเพียงว่าข้อมูลลำดับ 9164761 ไม่ได้รับ จึงส่งใหม่เพียง Packet เดียว และส่งข้อมูลลำดับ 10255081 ต่อไปตามปกติ แต่ฝั่งรับตอบกลับมาด้วย Ack# = 9166081 ซึ่งเป็นข้อมูลลำดับถัดไปที่ฝั่งรับต้องการ ซึ่งเป็นการ Ack ด้วย Ack# = 9166081 เป็นครั้งแรก

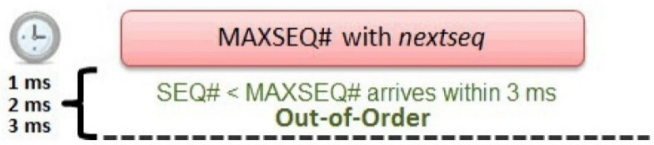
จึงได้ส่ง Packet 12038 ที่มีลำดับถัดไป คือ 10256401 แต่ฝั่งรับตอบกลับมาด้วยลำดับข้อมูลน้อยที่สุดที่ยังไม่ได้รับ คือ Ack# = 9166081 จึงถือเป็นการ Duplicate Ack และเกิดขึ้นจนถึง Packet 12247 รวมทั้งหมด 105 ครั้ง จึงได้มีการส่งข้อมูลลำดับ 9166081 ไปใน Packet 12248 จึงเป็นการ Retransmission อีกครั้ง

คำตอบ 1) ทราบเมื่อได้รับ Ack กลับมาที่มี ACK# ไม่ตรงตาม NEXT SEQ# คือตั้งแต่ 10418 หรือถ้านับจาก Duplicate ACK จำนวน 3 ครั้ง ก็จะเท่ากับ 10424
2) มีการส่งใหม่ โดยส่งใน Packet 12035 โดยใช้เวลา 0.465989 (นับจาก 10417) หรือ 0.476811 (นับจาก 10416) หรือ 0.000033 (นับจาก Dup ACK สุดท้าย)

Packet	SEQ#	NEXT SEQ#	ACK#	
10416	9163441	9164761		
10417	9175321	9176641		Previous segment not captured 10560 ไบต์
10418-11497	9176641	9889441	9164761	Dup ACK [#1 - #540]
11499	9901321	9902641		Previous segment not captured 11880 ไบต์
11501-12034	9902641	10255081	9164761	Dup ACK [#541 - #808]
12035	9164761	9166081		Fast Retransmission ของ 9164761
12036	10255081	10256401		
12037	1	1	9166081	Ack ของ 9166081 ครั้งแรก
12038	10256401	10257721	9166081	
12039-12247	1	1	9166081	Dup ACK [#1 - #105] ของ 9166081 #Time = 0
12248	9166081	9167401		Fast Retransmission ของ 9166081

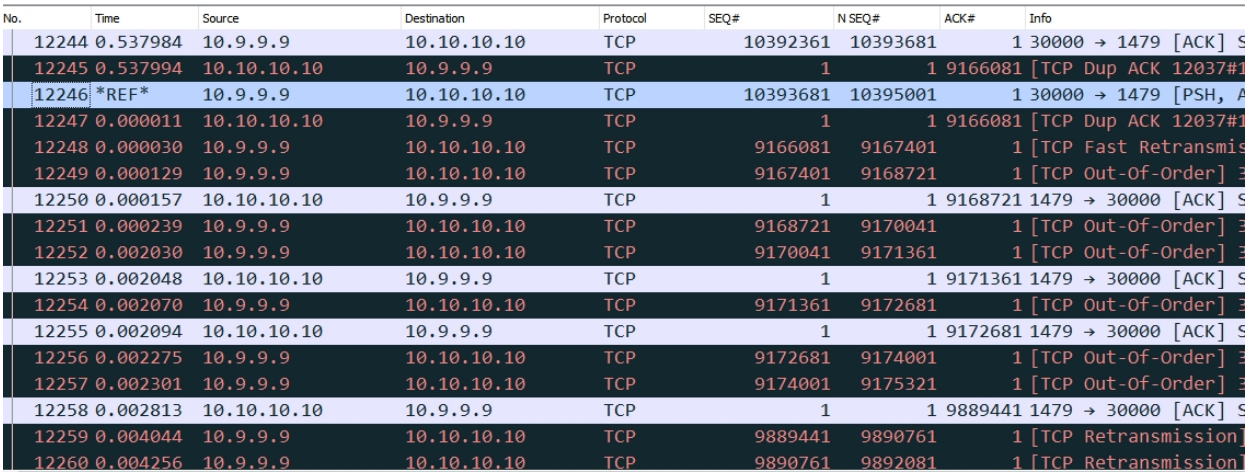
5. ให้ใช้ Display Filter : tcp.analysis.out_of_order จะพบ Out of Order อยู่ 8 ครั้ง ให้หาว่า Packet 12249 เป็น Out of Order จากเหตุการณ์หรือเงื่อนไขใด ให้ศึกษาวิธีการที่ Wireshark ตัดสินใจและอธิบายโดยย่อ
และในเวลาต่อมาที่ Packet 12249 มีการส่งข้อมูลลำดับที่ 9167401 ซึ่งต่อเนื่องจาก Packet 12248 (ซึ่งต่างจาก Packet 12036 ที่กลับไปส่งตามลำดับปัจจุบัน) จากนั้นฝั่งรับส่ง ACK# 9168721 กลับมา ฝั่งส่งจึงได้ส่งข้อมูลที่หายไปจนครบ ใน Packet ที่ 12251, 12252, 12254, 12256 และ 12257
จนถึง Packet 12258 จึงได้ส่ง ACK# 9889441 ซึ่งเป็นข้อมูลที่ยังไม่ได้รับที่น้อยที่สุด

แต่สำหรับสาเหตุที่ Wireshark ตีความว่าเป็น out of order เนื่องจาก Packet ทั้ง 6 มี SEQ# < 10395001 ซึ่งเป็นค่าที่น้อยกว่า ACK# สูงสุดที่เคยได้รับ (ลำดับ 10395001 ใน Packet 12246) และอยู่ในช่วงเวลา 3 ms จาก Packet 12246_



เมื่อกำหนด Time Reference = 0 ให้กับ Packet 12246 จะเห็นว่า Packet ที่แสดงเป็น Out of Order ทั้งหมด อยู่ในช่วงเวลา 3 ms ทั้งสิ้น สรุปว่า Packet 12249 เป็น Out of Order ของ Segment 12246

Packet	SEQ#	NEXT SEQ#	ACK#	
10416	9163441	9164761		
10417	9175321	9176641		Previous segment not captured 10560 ไบต์
10418-11497	9176641	9889441	9164761	Dup ACK [#1 - #540]
11499	9901321	9902641		Previous segment not captured 11880 ไบต์
11501-12034	9902641	10255081	9164761	Dup ACK [#541 - #808]
12035	9164761	9166081		Fast Retransmission ของ 9164761
12036	10255081	10256401		
12037	1	1	9166081	Ack ของ 9166081 ครั้งแรก
12038	10256401	10257721	9166081	
12039-12247	1	1	9166081	Dup ACK [#1 - #105] ของ 9166081 #Time = 0
12248	9166081	9167401		Fast Retransmission ของ 9166081
12249-12257	9167401	9175321		Out-of-order #Time < 3 ms
12258	1	1	9889441	



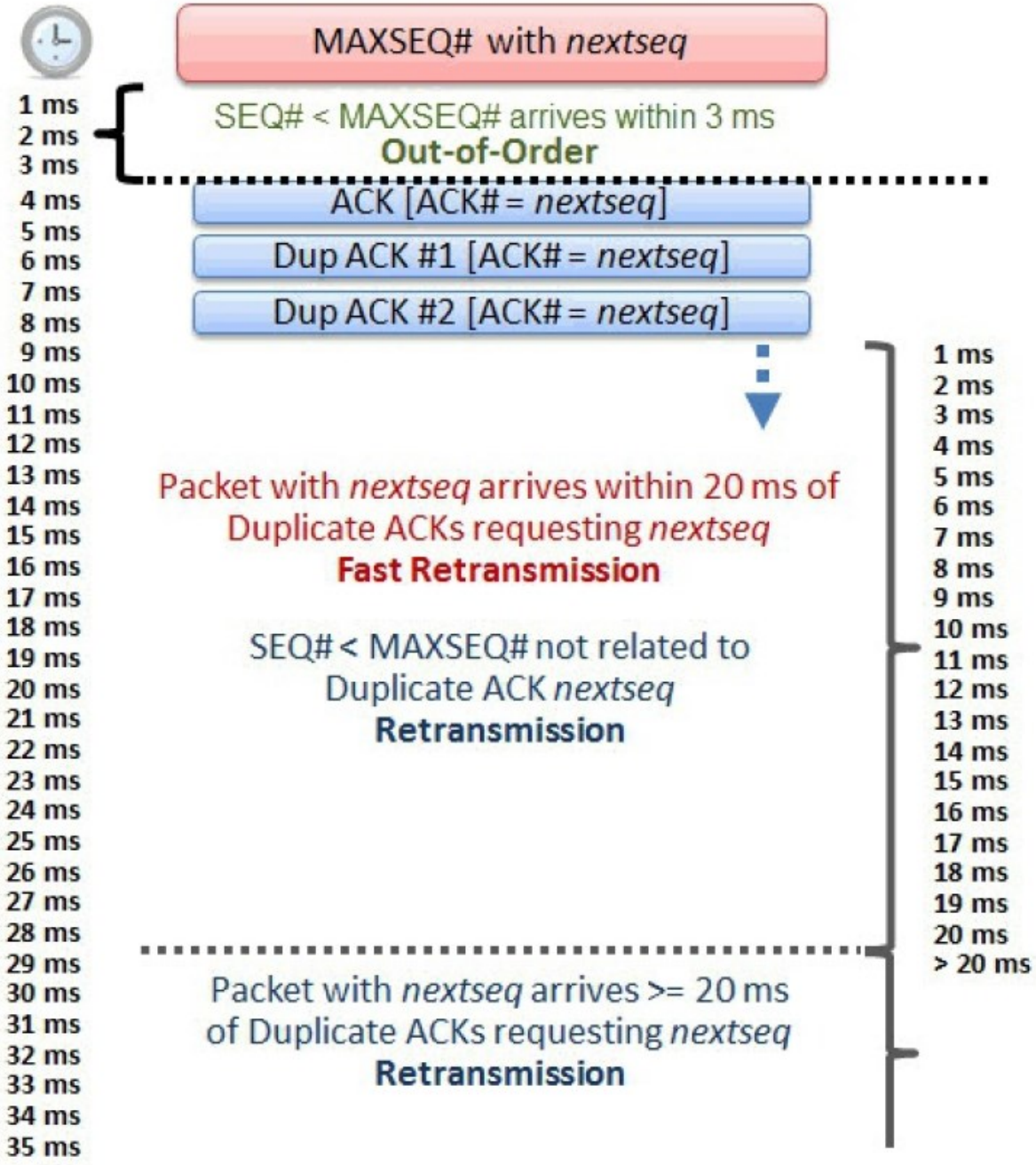
No.	Time	Source	Destination	Protocol	SEQ#	N SEQ#	ACK#	Info
12244	0.537984	10.9.9.9	10.10.10.10	TCP	10392361	10393681	1	30000 → 1479 [ACK] S
12245	0.537994	10.10.10.10	10.9.9.9	TCP	1	1	9166081	[TCP Dup ACK 12037#1
12246	*REF*	10.9.9.9	10.10.10.10	TCP	10393681	10395001	1	30000 → 1479 [PSH, A
12247	0.000011	10.10.10.10	10.9.9.9	TCP	1	1	9166081	[TCP Dup ACK 12037#1
12248	0.000030	10.9.9.9	10.10.10.10	TCP	9166081	9167401	1	[TCP Fast Retransmis
12249	0.000129	10.9.9.9	10.10.10.10	TCP	9167401	9168721	1	[TCP Out-Of-Order] 3
12250	0.000157	10.10.10.10	10.9.9.9	TCP	1	1	9168721	1479 → 30000 [ACK] S
12251	0.000239	10.9.9.9	10.10.10.10	TCP	9168721	9170041	1	[TCP Out-Of-Order] 3
12252	0.002030	10.9.9.9	10.10.10.10	TCP	9170041	9171361	1	[TCP Out-Of-Order] 3
12253	0.002048	10.10.10.10	10.9.9.9	TCP	1	1	9171361	1479 → 30000 [ACK] S
12254	0.002070	10.9.9.9	10.10.10.10	TCP	9171361	9172681	1	[TCP Out-Of-Order] 3
12255	0.002094	10.10.10.10	10.9.9.9	TCP	1	1	9172681	1479 → 30000 [ACK] S
12256	0.002275	10.9.9.9	10.10.10.10	TCP	9172681	9174001	1	[TCP Out-Of-Order] 3
12257	0.002301	10.9.9.9	10.10.10.10	TCP	9174001	9175321	1	[TCP Out-Of-Order] 3
12258	0.002813	10.10.10.10	10.9.9.9	TCP	1	1	9889441	1479 → 30000 [ACK] S
12259	0.004044	10.9.9.9	10.10.10.10	TCP	9889441	9890761	1	[TCP Retransmission]
12260	0.004256	10.9.9.9	10.10.10.10	TCP	9890761	9892081	1	[TCP Retransmission]

6. ไปที่ packet 12259 จะพบว่าเป็น Retransmission ให้บอกว่าเป็น Retransmission จาก RTO Timer หรือจากการได้รับ 3 Duplicate Ack พร้อมเหตุผลประกอบโดยย่อ

   ก่อนอื่นต้องเข้าใจว่า Retransmission Timer นั้นเป็นค่าที่เปลี่ยนแปลงระหว่างที่คอมพิวเตอร์ทำงาน แต่เนื่องจากการ Capture ของ Packet Capture Software นั้น ไม่ทราบข้อมูลจริง ดังนั้น Wireshark จึงกำหนดอัลกอริทึมของตนเองขึ้นมา โดยกำหนดเงื่อนไขว่า หาก Retransmission เกิดขึ้นภายในเวลา 20 ms นับจากที่ครบ 3 Duplicate ACK (ดูรูปประกอบ) จะถือว่าเป็น Fast Retransmission และหากเกินเวลาก็จะถือว่าเป็น Retransmission

   ใน Packet 12259 เป็นการส่ง Retransmission ของ Packet ที่หายไประหว่าง 11497 และ 11499 ซึ่งมีจำนวน 9 Packet และหากกำหนด Time Reference = 0 ให้กับ Packet 12043 ซึ่งเป็น Duplicate ACK ที่ 3 พบว่าจะจาก 12043 จนถึง 12259 ใช้เวลา 0.06486 วินาที ซึ่งเกินเวลามากกว่า 20 ms (0.02) ดังนั้นกรณีนี้ Wireshark จึงแสดงเป็น Retransmission

   อย่างไรก็ตามในความเป็นจริงแล้วเราไม่ทราบว่าการที่มีการส่งใหม่นั้นเป็นการส่งใหม่จากเงื่อนไขใด เพราะเป็นโปรแกรมที่เขียนโดยผู้พัฒนา OS


MAXSEQ# with nextseq
1 ms
2 ms
3 ms
SEQ# < MAXSEQ# arrives within 3 ms
Out-of-Order
4 ms
5 ms
6 ms
7 ms
8 ms
9 ms
10 ms
11 ms
12 ms
13 ms
14 ms
15 ms
16 ms
17 ms
18 ms
19 ms
20 ms
21 ms
22 ms
23 ms
24 ms
25 ms
26 ms
27 ms
28 ms
29 ms
30 ms
31 ms
32 ms
33 ms
34 ms
35 ms
ACK [ACK# = nextseq]
Dup ACK #1 [ACK# = nextseq]
Dup ACK #2 [ACK# = nextseq]
Packet with nextseq arrives within 20 ms of Duplicate ACKs requesting nextseq
Fast Retransmission
SEQ# < MAXSEQ# not related to Duplicate ACK nextseq
Retransmission
1 ms
2 ms
3 ms
4 ms
5 ms
6 ms
7 ms
8 ms
9 ms
10 ms
11 ms
12 ms
13 ms
14 ms
15 ms
16 ms
17 ms
18 ms
19 ms
20 ms
> 20 ms
Packet with nextseq arrives >= 20 ms of Duplicate ACKs requesting nextseq
Retransmission