

Phân tích hoạt động giao thức TCP - UDP

TCP/UDP Protocol

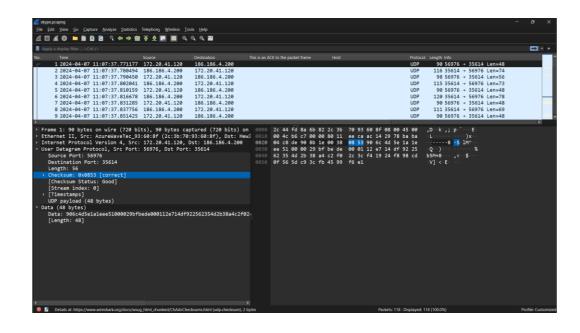
1. Tính checksum của 1 gói tin UDP bất kỳ (dựa trên thông tin bắt được từ wireshark checksum

Trả lời:

	Decimal	Binary	Hex
Source IP Destination IP	192.168	1100 0000 1010 1000	CO A8
	0.31	0000 0000 0001 1111	00 1F
	192.168.	1100 0000 1010 1000	CO AS
	0.30	0000 0000 0001 1110	00 18
Reserved/UDP	0/17	0000 0000 0001 1110	00 11
protocol			8
Padding/Length	0/10	0000 0000 0000 1010	00 0
Pseudo header ends h	ere so we v	will add the real UDP header to tl	nis
UDP Source Port	20	0000 0000 0001 0100	00 14
UDP destination Port	10	0000 0000 0000 1010	00 04
UDP Length	10	0000 0000 0000 1010	00 00
UDP Data	Hi	0100 1000 0110 1001	48 69
Now that we have all	that inform	ation late and d	
Now that we have all	mar imorm	iation let's add	
		1 1100 1010 0011 1001	
Notice in our previous since our checksum ha from t to become 32 b	s entry our v as to be 16 oits. Thus w	CONTROL MAN CONTROL CO	and the results
Notice in our previous since our checksum ha from t to become 32 b	s entry our v as to be 16 oits. Thus w	1 1100 1010 0011 1001 values exceed 16 bits (2 bytes). T bits. To get to 16 bits we will exp e will prepend hex 000 to 1 CA 39	his will not work and the results
Notice in our previous since our checksum ha from t to become 32 b find the binary value o	entry our vas to be 16 oits. Thus word of 000 and a	1 1100 1010 0011 1001 values exceed 16 bits (2 bytes). T bits. To get to 16 bits we will exp e will prepend hex 000 to 1 CA 39 add it to the binary column.	his will not work and the results 9. We will also 00 01 CA 39
Notice in our previous since our checksum ha from t to become 32 the find the binary value of the Now that we have the	entry our vas to be 16 oits. Thus word of 000 and a	1 1100 1010 0011 1001 values exceed 16 bits (2 bytes). T bits. To get to 16 bits we will exp e will prepend hex 000 to 1 CA 39 add it to the binary column. 1 1100 1010 0011 1001	his will not work and the results 9. We will also 00 01 CA 39
Notice in our previous since our checksum ha from t to become 32 the find the binary value of the Now that we have the	entry our vas to be 16 oits. Thus word of 000 and a	1 1100 1010 0011 1001 values exceed 16 bits (2 bytes). T bits. To get to 16 bits we will exp e will prepend hex 000 to 1 CA 39 add it to the binary column. 1 1100 1010 0011 1001 he we take the upper half 00 01 a	his will not work and the results 9. We will also 00 01 CA 39 nd add them to
Notice in our previous since our checksum ha from t to become 32 the find the binary value of the Now that we have the	entry our vas to be 16 oits. Thus word of 000 and a	1 1100 1010 0011 1001 values exceed 16 bits (2 bytes). T bits. To get to 16 bits we will exp e will prepend hex 000 to 1 CA 39 add it to the binary column. 1 1100 1010 0011 1001 the we take the upper half 00 01 a	his will not work and the results 9. We will also 00 01 CA 39 nd add them to 00 00 + CA 39
Notice in our previous since our checksum hat from t to become 32 to find the binary value of the lower half CA 39 We're getting there.	s entry our vas to be 16 bits. Thus wor of 000 and a 2 32 Bit value.	1 1100 1010 0011 1001 values exceed 16 bits (2 bytes). T bits. To get to 16 bits we will exp e will prepend hex 000 to 1 CA 39 add it to the binary column. 1 1100 1010 0011 1001 te we take the upper half 00 01 a 0000 0000 0000 0001 + 1100 1010 0011 1001	his will not work and the results 9. We will also 00 01 CA 33 nd add them to 00 01 + CA 33 CA34 to find its one's

Ta dựa vào bảng, ta có thể thiết lập công thức.

PS C:\Users\DucAnh> python -u "c:\Users\DucAnh\OneDrive\Documents\Study\UIT\hk2\NMMT\TH\Lab 3\checksum.py"
Input Ip of Source: 172.20.41.120
Input Ip of Destination: 186.186.4.200
Input port of Source: 56976
Input port of Destination: 35614
Input length: 56
Input payload: 906c4d5e1a1eee51000029bfbede000112e714df922562354d2b38a4c2f02c3cf41924f898cd0f565dc93cfb4599f6e1
Check sum: 0x0853



2. Tìm gói tin tương ứng với các sự kiện sau:

TCP sender

- Sự kiện: Nhận dữ liệu từ tầng ứng dụng

- Sự kiện: timeout

- Sự kiện: nhận được ACK

TCP Receiver: ACK generation [RFC 5681]

- Nhận được segment đúng với STT đang chờ. Tất cả segment trước đó đã được ACK.
- Nhận được segment đúng với STT đang chờ, một segment chưa được ACK.
- Nhận được segment không đúng thứ tự (STT cao hơn).
- Nhận được segment trong khoảng bị trống (giữa STT đang chờ và STT nhận được trước đó).

Trả lời:

TCP sender

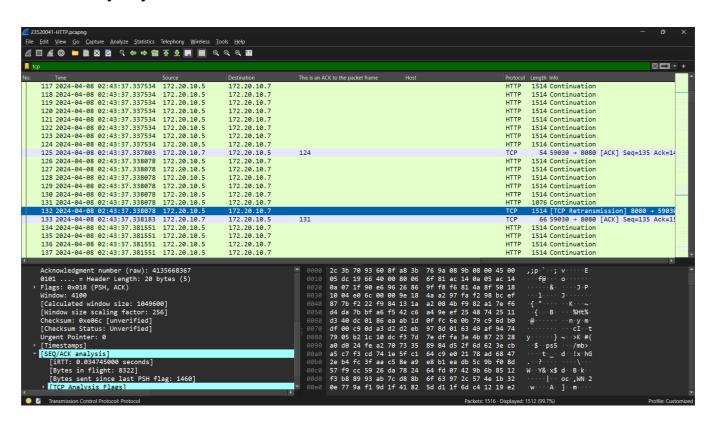
- Sự kiện: nhận dữ liệu từ tầng ứng dụng

o Sử dụng filter data.data **⊠** □ + 35 2024-04-08 02:43:36.73682 39 2024-04-08 02:43:36.753652 39 2024-04-08 02:43:36.753652 41 2024-04-08 02:43:36.753652 41 2024-04-08 02:43:36.757653 43 2024-04-08 02:43:36.757663 42 2024-04-08 02:43:36.757663 45 2024-04-08 02:43:36.757663 47 2024-04-08 02:43:36.760919 172.20.10.5 172.20.10.7 1514 Continuation 47 2024-04-08 02:43:36.760919
48 2024-04-08 02:43:36.760919
49 2024-04-08 02:43:36.760919
51 2024-04-08 02:43:36.760919
51 2024-04-08 02:43:36.760919
52 2024-04-08 02:43:36.760919
55 2024-04-08 02:43:36.760919
55 2024-04-08 02:43:36.850563
57 2024-04-08 02:43:36.850563
57 2024-04-08 02:43:36.850563
58 2024-04-08 02:43:36.850563 172.20.10.5 172.20.10.7 1514 Continuation 172.20.10.5 1514 Continuation 172.20.10.5 172.20.10.7 1514 Continuation 172.20.10.5 172.20.10.5 172.20.10.5 172.20.10.5 172.20.10.5 172.20.10.5 172.20.10.5 1514 Continuation 1514 Continuation 1514 Continuation 1514 Continuation 1514 Continuation 1514 Continuation 2c 3b 70 93 60 8f 7e 55 87 41 b7 a5 08 09 45 66
05 dc 19 11 40 00 80 05 6f d6 ac 14 0a 05 ac 14
0a 07 1f 90 e6 96 26 84 d3 37 f6 81 4a 8f 50 10
10 04 ce 41 00 00 74 69 0f 49 22 9a ac cf 59 f6
08 1f 39 ec 92 74 94 d3 93 74 ad 79 66 51 7b
73 b7 42 a9 26 84 20 7d ff 48 24 f5 3a 97 6e 44
45 fc 02 2c 51 ad b0 e8 cd d9 15 76 64 26 af ea ac
60 7b 6f 40 a2 1d 00 62 14 0f 6b 62 d3 69 54 d1
2b 93 18 8d f6 75 81 72 d6 df 11 85 86 b9 a1 e3
df c3 99 6f 36 27 63 of 30 83 07 07 73 be 26 8a
a6 92 96 77 3e 08 a8 54 e9 d4 e2 76 3a f4 a6 ff
8e 90 ef 0c a5 f2 e4 18 0b 3a 11 1c c6 de 21 dd
46 56 b9 18 73 a5 4e 4d 86 80 55 ec 32 3e 3a 8f
5e 5e f4 bc f5 bf 9c 23 0a ca c8 3b 64 47 b4 ed
0d 88 00 52 b8 f4 55 75 7d 19 4f 75 0c c5 85 90
3c b6 ee f8 60 9a 8f 00 8e fe 1b 84 d0 48 7a 3c
c5 e2 30 91 7d c3 f4 68 55 68 03 10 f1 be d1 b1 Frame 36: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 Ethernet II, Src: 7e:55:87:41:b7:a5 (7e:55:87:41:b7:a5), Dst: AzureWax Internet Protocol Version 4, Src: 172.20.10.5, Dst: 172.20.10.7 Transmission Control Protocol, Src Port: 8080, Dst Port: 59030, Seq: E Hypertext Transfer Protocol t tymQ{ } H\$: nD s B & E ,Q {o@ FV s NM U 2>:

^^ # ;dG

R Uu } Ou

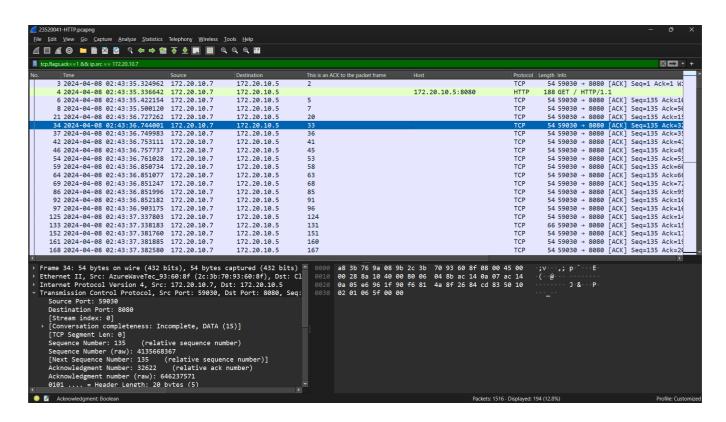
- Sự kiện: timeout



```
## Site | Formation | State |
```

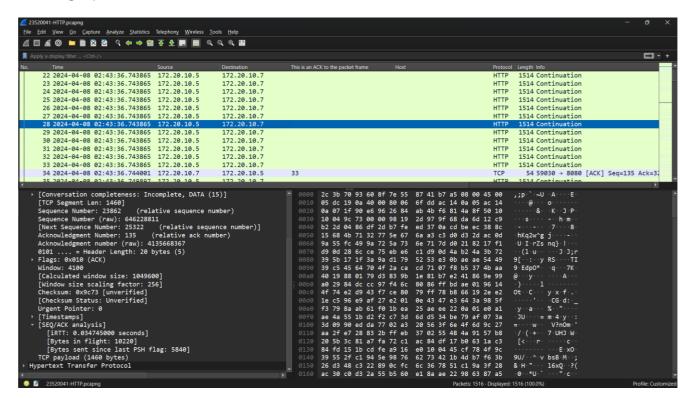
- Sự kiện: nhận được ACK

O Sử dụng filter: tcp.flags.ack==1 && ip.src == 172.20.10.7



TCP receiver

- Nhận được segment đúng với STT đang chờ. Tất cả segment trước đó đã được ACK.



- Nhận được segment đúng với STT đang chờ, một segment chưa được ACK.

Không có

- Nhận được segment không đúng thứ tự (STT cao hơn).

Không có

- Nhận được segment trong khoảng bị trống (giữa STT đang chờ và STT nhận được trước đó)

Không có