

25 March 2025

Spring 2025

(INFT-3508 - 20332)

Cyber Security Fundamentals

Homework 1

Task 1

First, I observed that the text in “book.pdf” is divided into paragraphs. Since the highest first digit in the number sets is 8 (as seen in (8,10,1)), there are at least 8 paragraphs in the document. This means the first number in each triplet likely refers to a specific paragraph. For example, in (1,9,4), the “1” points to the first paragraph.

Next, the second number in each triplet probably indicates the line within the paragraph. Paragraphs are made up of lines, and these line numbers seem to fit logically. For instance, in (1,9,4), the “9” refers to the ninth line of the first paragraph.

Finally, the third number in the triplet specifies the position of a word within that line. For example, in (1,9,4), the “4” means the fourth word in the ninth line of the first paragraph. Similarly, (7,1,5) would point to the fifth word in the first line of the seventh paragraph.

By following this pattern – (paragraph number, line number, word position), we can extract a specific hidden message from the “book.pdf” document. This message is **“The flag is Ceremonial plates”**.

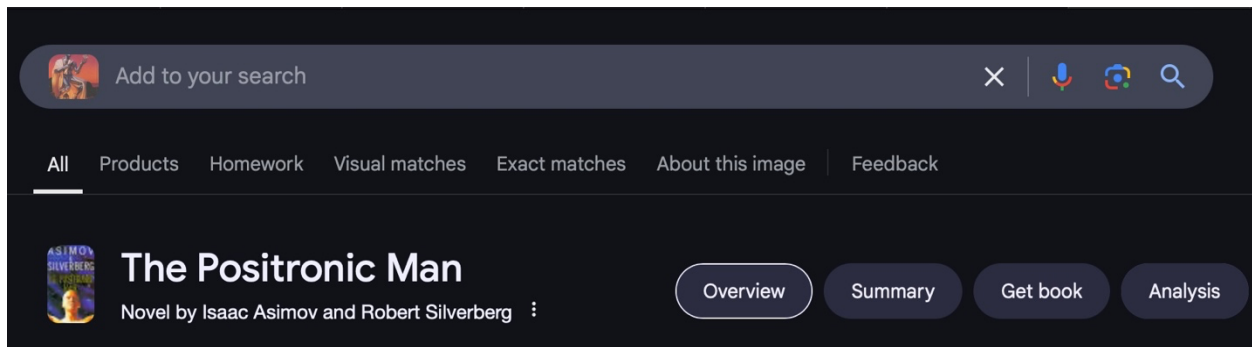
(1,9,4) → 1st paragraph, 9th line, 4th word

It might have been expected that the attempt to trace to their origin in the past the institutions and customs in common use upon the sea would from an early date occupy the attention of a seafaring people, but for some obscure reason the British nation has always been indifferent to the history of its activities upon that element on which its greatness was founded, and to which it has become more and more dependent for its daily bread and its very existence. To those who are alive to this fact it will hardly come as a surprise, therefore, to learn that the first sustained attempt at a detailed investigation into the history of the flag at sea was made under the patronage of the German Admiralty by a German Admiral. Vice-Admiral Siegel's Die Flagge, published in 1912, was the first book to deal with the development of the flag at sea in a scientific spirit, and although the earlier chapters contain some mistakes due to his employment of translations of early works instead of original texts, and the accounts of the British flags in the later chapters suffer because he had no access to original records, it is a worthy piece of work.

Task 2

To solve this puzzle, we need to identify the name of the book depicted in the poster. Firstly, I downloaded the poster and uploaded the image to a search bar on Google. This technique is called Reverse Image Search. It works by analyzing the visual patterns in an uploaded image, comparing it against millions of images online, and identifying matching or similar images along with their associated web pages.

After this step, I found that this poster is from a science fiction novel about robots and humanity. The name of the book is “The Positronic Man” by Isaac Asimov and Robert Silverberg.



Task 3

The captured network traffic reveals a sequence of communications between the local device (IP: 172.16.1.132) and different remote servers. Initially, a TCP handshake takes place, which is a standard three-step process for establishing a reliable connection. The client first sends a SYN packet, the server responds with a SYN-ACK, and the client completes the process with an ACK. This confirms that both devices are ready to exchange data.

After establishing the connection, multiple HTTP GET and POST requests follow. The GET requests indicate that the client is fetching resources from a website, likely for loading scripts and other webpage components. An essential detail in this traffic capture is an HTTP POST request that contains sensitive information. Since HTTP does not encrypt data, credentials are visible in plain text.

First Attempt

139	17.231994	172.16.1.132	104.131.53.208	HTTP	100	POST /login/ HTTP/1.1 (application/x-www-form-urlencoded)
140	17.283799	104.131.53.208	172.16.1.132	TCP	66	8080 → 55720 [ACK] Seq=1 Ack=485 Win=30208 Len=0 TSval=44107408 TSecr=458487600
141	17.284296	104.131.53.208	172.16.1.132	TCP	66	8080 → 55720 [ACK] Seq=1 Ack=519 Win=30208 Len=0 TSval=44107408 TSecr=458487601
142	17.285606	104.131.53.208	172.16.1.132	TCP	83	8080 → 55720 [PSH, ACK] Seq=1 Ack=519 Win=30208 Len=17 TSval=44107409 TSecr=458487601 [TCP PDU reas
143	17.285642	172.16.1.132	104.131.53.208	TCP	66	55720 → 8080 [ACK] Seq=519 Ack=18 Win=131296 Len=0 TSval=458487653 TSecr=44107409
144	17.286484	104.131.53.208	172.16.1.132	HTTP	865	HTTP/1.0 200 OK (text/html)
145	17.286541	172.16.1.132	104.131.53.208	TCP	66	55720 → 8080 [ACK] Seq=519 Ack=818 Win=130512 Len=0 TSval=458487654 TSecr=44107409
146	17.287018	172.16.1.132	104.131.53.208	TCP	66	55720 → 8080 [FIN, ACK] Seq=519 Ack=818 Win=131072 Len=0 TSval=458487654 TSecr=44107409
147	17.339142	104.131.53.208	172.16.1.132	TCP	66	8080 → 55720 [ACK] Seq=818 Ack=520 Win=30208 Len=0 TSval=44107422 TSecr=458487654
148	24.822691	172.16.1.132	104.131.53.208	TCP	78	55721 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=16 TSval=458495182 TSecr=0 SACK_PERM
149	24.875508	104.131.53.208	172.16.1.132	TCP	74	8080 → 55721 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1380 SACK_PERM TSval=44109306 TSecr=4584951

Acknowledgment number (raw): 1806774958	0000	f8 c2 88 1b 5f d4 80 e6	50 1d 18 e4 08 00 45 00	P E .
1000 = Header Length: 32 bytes (8)	0010	00 78 54 f1 40 00 40 06	99 a7 ac 10 01 84 68 83	V : @ h
Flags: 0x018 (PSH, ACK)	0020	35 d0 d9 a9 1f 90 a3 a7	4c 06 6b b1 32 ae 80 18	5 L . k . 2 . . .
Window: 8208	0030	20 10 5b f1 00 00 01 01	08 0a 1b 54 15 03 02 a1	[. S . 1 . . .
[Calculated window size: 131328]	0040	0d fa 75 73 65 72 6e 61	6d 65 3d 63 6c 61 75 64	. : u s e r n a m e = c l a u d i o
[Window size scaling factor: 16]	0050	69 6f 26 70 61 73 73 77	6f 72 64 3d 66 6c 61 67	i o 6 p a s s w o r d = p a s s w o r d
Checksum: 0x8bb3 [unverified]	0060	25 32 34 25 37 44		
[Checksum Status: Unverified]				
Urgent Pointer: 0				

Second Attempt

152	24.876627	172.16.1.132	104.131.53.208	HTTP	134	POST /login/ HTTP/1.1 (application/x-www-form-urlencoded)
153	24.930302	104.131.53.208	172.16.1.132	TCP	66	8080 → 55721 [ACK] Seq=1 Ack=485 Win=30208 Len=0 TSval=44109320 TSecr=458495235
154	24.930338	104.131.53.208	172.16.1.132	TCP	66	8080 → 55721 [ACK] Seq=1 Ack=553 Win=30208 Len=0 TSval=44109320 TSecr=458495235
155	24.935727	104.131.53.208	172.16.1.132	TCP	83	8080 → 55721 [PSH, ACK] Seq=1 Ack=553 Win=30208 Len=17 TSval=44109321 TSecr=458495235 [TCP PDU
156	24.935765	104.131.53.208	172.16.1.132	HTTP	598	HTTP/1.0 200 OK (text/html)
157	24.935824	172.16.1.132	104.131.53.208	TCP	66	55721 → 8080 [ACK] Seq=553 Ack=18 Win=131296 Len=0 TSval=458495293 TSecr=44109321
158	24.935825	172.16.1.132	104.131.53.208	TCP	66	55721 → 8080 [ACK] Seq=553 Ack=551 Win=130768 Len=0 TSval=458495293 TSecr=44109321

Sequence Number (raw): 3933835839	0000	f8 c2 88 1b 5f d4 80 e6	50 1d 18 e4 08 00 45 00	P E .
[Next Sequence Number: 553 (relative sequence number)]	0010	00 78 54 f1 40 00 40 06	99 a7 ac 10 01 84 68 83	x T : @ h
Acknowledgment Number: 1 (relative ack number)	0020	35 d0 d9 a9 1f 90 ea 79	92 3f 68 66 1a 98 80 18	5 y . 7 h f
Acknowledgment number (raw): 1751521944	0030	20 10 5b f1 00 00 01 01	08 0a 1b 54 15 03 02 a1	[. S . 1 . . .
1000 = Header Length: 32 bytes (8)	0040	0d fa 75 73 65 72 6e 61	6d 65 3d 63 6c 61 75 64	. : u s e r n a m e = c l a u d i o
Flags: 0x018 (PSH, ACK)	0050	69 6f 26 70 61 73 73 77	6f 72 64 3d 66 6c 61 67	i o 6 p a s s w o r d = f l a g % 7 B p l % 2 4 _ % 2 4 % 2 4 l _ y 0 u r _ l o g i n _ f o r m % 2 4 % 7 D
Window: 8208	0060	25 32 34 25 37 44		
[Calculated window size: 131328]				
[Window size scaling factor: 16]				
Checksum: 0x5f61 [unverified]				
[Checksum Status: Unverified]				
Urgent Pointer: 0				

These pictures represent an HTTP POST request to “/login,” indicating that a user is attempting to log into a website or system by sending their username and password through this request to the “/login” endpoint. The content type is “application/x-www-form-urlencoded,” which signifies that it transmits form data, including a username and password.

Here, we can see two login attempts. The first attempt is in packet 139, where the password is “password,” which is incorrect. After this failed attempt, there is a second login attempt in packet 152. When we click on the packet, we can clearly see both the username and the correct password in the right-bottom section. (highlighted with orange color)

- Username: claudio
- Password: flag%7Bpl%24_%24%24l_y0ur_l0g1n_form%24%7D

If we decode the existing password in hexadecimal values, we get the password:
redflag{p1\$_\$\$l_y0ur_l0g1n_form\$}

Task 4

In this task, we are trying to decode an encrypted text. The structure of the text looks like a song, but at the top, it seems like there is a separate sentence. This sentence is likely important; therefore, I decided to focus on decoding it first.

First, I looked for short, frequently repeated words in the text. I found “wcn,” which appears many times throughout the text and starts the first sentence and song. Since “the” is a very common word (article) in English and often appears at the beginning of sentences, I guessed that “wcn” might mean “the.” In this case “w” is “t”, “c” is “h”, and “n” is “e”. To know that this way is true I started testing on different words and started decoding.

Next, I searched for other short words, such as prepositions and articles, because they often follow predictable patterns. By identifying these words, I could decode more letters. Each time I decoded a new letter, I wrote it down, allowing me to apply it to other words and gradually reveal them. This step-by-step process made it easier to decode the rest of the text.

1. aw (3rd line 3rd word) - _ t → can be “it”
2. at (28th line) – i _ → can be “if”
3. ht (14th line 7 th word) - _ f → can be “of”
4. enn (4th line 5th word) - _ ee → can be “see”
5. cze (11th line 1st word) - _ as → can be “has”
6. oatn (11th line 3rd word) - _ ife → can be “life”
7. oayn (13th line 8th word) – li _ e → can be “like”
8. wcady (1st line 2nd word) – thi _ k → can be “think”
9. ewaoo (4th line 2nd word) – sti _ _ → “still”
10. zoo (18th line 5th word) – a _ _ → can be “all”
11. jawc (18th line 4th word) - _ ith → can be “with”
12. zdg (2nd line 1st word) – an _ → can be “and”
13. thvnew - (21st line 9th word) – fo _ est → can be “forest”
14. ihohve - (19th line 7th word) - _ olors → can be “colors”
15. izd (9th line 9th word) - _ an → can be “can”
16. dzrn - (11th line 9th word) – na _ e → can be “name”
17. rhkdwzade (18th line 10th word) – mo _ ntains → can be “mountains”

As I noted the letters, it looked like this:

w-t
c-h
n-e
z (is used very often and separately) - a
a-i
t-f
h-o
e-s
o-L
y-k
j-w
d-n
g-d
v-r
r-m
i-c
k-u

With these letters identified, I could finally decode the first sentence “**wcn
zkwchvapzwahd ihgn ae jawczoowcnihohvehtwcnjadg**”

- wcn = “the”
- zkwchvapzwahd = “authorization” (Since “p” was not yet decoded, I identified the missing letter logically as “z”)
- ihgn = “code”
- ae = “is”
- jawczoowcnihohvehtwcnjadg = “withallthecolorsofthewind”

By putting all the decoded words together, I found the full sentence: “**The authorization code is withallthecolorsofthewind.**” From this sentence, it is immediately clear that the actual authorization code is “**withallthecolorsofthewind.**”

Task 5

In order to determine the attacker's last name, we need to carefully analyze packets. One key observation is in packet 58, where we see an HTTP GET request for a file named "secretfile.txt." The source IP address is 192.168.50.4, and the destination is 192.168.50.10. However, there is a suspicious activity. Before the broadcast, in packet 51 we can see that the expected destination is 192.168.50.5, not 192.168.50.4.

50	190.664537	192.168.50.5	192.168.50.10	TCP	66	58996 → 80 [FIN, ACK] Seq=101 Ack=633 Win=15872 Len=0 TSval=429355 TSecr=410164
51	190.664863	192.168.50.10	192.168.50.5	TCP	66	80 → 58996 [ACK] Seq=633 Ack=102 Win=14480 Len=0 TSval=410164 TSecr=429355
52	533.426706	192.168.50.1	192.168.50.255	DB-LS	210	Dropbox LAN sync Discovery Protocol, JSON
53	539.439371	PCSSystemtec_2b:f7...	Broadcast	ARP	42	Who has 192.168.50.10? Tell 192.168.50.4 (duplicate use of 192.168.50.4 detected!)
54	539.439891	PCSSystemtec_a3:7c...	PCSSystemtec_2b:f7...	ARP	60	192.168.50.10 is at 08:00:27:a3:7c:ac (duplicate use of 192.168.50.4 detected!)
55	539.439939	192.168.50.4	192.168.50.10	TCP	74	51064 → 80 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=550129 TSecr=0 WS=16
56	539.440360	192.168.50.10	192.168.50.4	TCP	74	80 → 51064 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM TSval=497357 TSecr=550129
57	539.440458	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=550129 TSecr=497357
58	539.440695	192.168.50.4	192.168.50.10	HTTP	165	GET /secretfile.txt HTTP/1.1
59	539.441119	192.168.50.10	192.168.50.4	TCP	66	80 → 51064 [ACK] Seq=1 Ack=100 Win=14480 Len=0 TSval=497357 TSecr=550129
60	539.441719	192.168.50.10	192.168.50.4	HTTP	590	HTTP/1.1 200 OK (text/plain)
61	539.441785	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [ACK] Seq=100 Ack=525 Win=15680 Len=0 TSval=550129 TSecr=497357
62	539.442160	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [FIN, ACK] Seq=100 Ack=525 Win=15680 Len=0 TSval=550129 TSecr=497357
63	539.468186	192.168.50.10	192.168.50.4	TCP	66	80 → 51064 [FIN, ACK] Seq=525 Ack=101 Win=14480 Len=0 TSval=497364 TSecr=550129
64	539.468228	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [ACK] Seq=101 Ack=526 Win=15680 Len=0 TSval=550136 TSecr=497364

> Frame 58: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits)	0000	08 00 27 a3 7c ac 08 00 27 2b f7 02 08 00 45 00+....E
> Ethernet II, Src: PCSSystemtec_2b:f7:02 (08:00:27:2b:f7:02), Dst: PCSSystemtec_a3:7c:ac	0010	00 97 d8 d3 40 00 40 06 7c 2e c0 a8 32 04 c0 a8	...@_ ...2...
> Destination: PCSSystemtec_a3:7c:ac (08:00:27:a3:7c:ac)	0020	32 0a c7 78 00 50 01 cb bc 21 67 33 d3 4a 08 18	2..x.P...lg3.J...
> Source: PCSSystemtec_2b:f7:02 (08:00:27:2b:f7:02)	0030	03 91 e5 e8 00 00 01 01 08 0a 00 08 64 f1 00 07	...d...
Type: IPv4 (0x0800)	0040	96 cd 47 45 54 20 2f 73 65 63 72 65 74 66 69 6c	...GET /s ecretfil
[Stream index: 4]	0050	65 2e 74 78 74 20 48 54 54 50 2f 31 2e 31 0d 0a	e.txt HT TP/1.1..
> Internet Protocol Version 4, Src: 192.168.50.4, Dst: 192.168.50.10	0060	55 73 65 72 2d 41 67 65 6e 74 3a 20 63 75 72 6c	User-Age nt: curl
> Transmission Control Protocol, Src Port: 51064, Dst Port: 80, Seq: 1, Ack: 1, Len: 99	0070	2f 37 2e 32 36 2e 30 0d 0a 48 6f 73 74 3a 20 76	/7.26.0 .Host: v
> Hypertext Transfer Protocol	0080	70 6e 2e 64 61 65 64 61 75 6c 75 73 63 6f 72 70	pn.daeda uluscorp
	0090	2e 63 6f 6d 0d 0a 41 63 63 65 70 74 3a 20 2a 2f	.com~Ac cept: */
	00a0	2a 0d 0a 0d 0a	*....

To uncover the actual IP address used by the attacker, we need to examine the last ARP request. The reason is that when an IP conflict or unauthorized device appears on the network, ARP requests help to detect the attack. By checking the last ARP request, we find a message "Duplicate use of 192.168.50.4 detected!". This confirms that an attacker is masquerading as 192.168.50.4.

Additionally, we identify that this attacker has a MAC address of **08:00:27:2b:f7:02**. This is a key piece of information because MAC addresses are unique in each device and cannot be easily changed like IP addresses.

52	533.426706	192.168.50.1	192.168.50.255	DB-LS	210	Dropbox LAN sync Discovery Protocol, JSON
53	539.439371	PCSSystemtec_2b:f7...	Broadcast	ARP	42	Who has 192.168.50.10? Tell 192.168.50.4 (duplicate use of 192.168.50.4 detected!)
54	539.439891	PCSSystemtec_a3:7c...	PCSSystemtec_2b:f7...	ARP	60	192.168.50.10 is at 08:00:27:a3:7c:ac (duplicate use of 192.168.50.4 detected!)
55	539.439939	192.168.50.4	192.168.50.10	TCP	74	51064 → 80 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM TSval=550129 TSecr=0 WS=16
56	539.440360	192.168.50.10	192.168.50.4	TCP	74	80 → 51064 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM TSval=497357 TSecr=550129
57	539.440458	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=550129 TSecr=497357
58	539.440695	192.168.50.4	192.168.50.10	HTTP	165	GET /secretfile.txt HTTP/1.1
59	539.441119	192.168.50.10	192.168.50.4	TCP	66	80 → 51064 [ACK] Seq=1 Ack=100 Win=14480 Len=0 TSval=497357 TSecr=550129
60	539.441719	192.168.50.10	192.168.50.4	HTTP	590	HTTP/1.1 200 OK (text/plain)
61	539.441785	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [ACK] Seq=100 Ack=525 Win=15680 Len=0 TSval=550129 TSecr=497357
62	539.442160	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [FIN, ACK] Seq=100 Ack=525 Win=15680 Len=0 TSval=550129 TSecr=497357
63	539.468186	192.168.50.10	192.168.50.4	TCP	66	80 → 51064 [FIN, ACK] Seq=525 Ack=101 Win=14480 Len=0 TSval=497364 TSecr=550129
64	539.468228	192.168.50.4	192.168.50.10	TCP	66	51064 → 80 [ACK] Seq=101 Ack=526 Win=15680 Len=0 TSval=550136 TSecr=497364

Frame 54: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)	0000	08 00 27 2b f7 02 08 00 27 a3 7c ac 08 06 00 01	...+... ...2...
> Ethernet II, Src: PCSSystemtec_a3:7c:ac (08:00:27:a3:7c:ac), Dst: PCSSystemtec_2b:f7:02	0010	08 00 06 04 00 02 08 00 27 a3 7c ac 08 32 0a	...+... ...2...
> Destination: PCSSystemtec_2b:f7:02 (08:00:27:2b:f7:02)	0020	08 00 27 2b f7 02 c0 a8 32 04 00 00 00 00 00	...+...2...
> Source: PCSSystemtec_a3:7c:ac (08:00:27:a3:7c:ac)	0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Type: ARP (0x0806)			
[Stream index: 4]			
Padding: 00000000000000000000000000000000			
Address Resolution Protocol (reply)			
[Duplicate IP address detected for 192.168.50.4 (08:00:27:2b:f7:02) - also in use by 08:			

Now, we need to analyze the user activity to find the attacker. Observing the packets, we can see that IP 192.168.50.3 requested /john.johnson endpoint. How do we know this is suspicious? The source MAC address (08:00:27:2b:f7:02) of this request matches the attacker's MAC address. This means that the attacker is pretending to be (masquerading) to be a real user.

Therefore, the answer is: **“johnson, 192.168.50.3, 192.168.50.4”**

20	52.115095	192.168.50.3	192.168.50.10	TCP	66	37291 → 80 [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=428298 TSecr=375524
21	52.115385	192.168.50.3	192.168.50.10	HTTP	165	GET /john.johnson HTTP/1.1
22	52.115812	192.168.50.10	192.168.50.3	TCP	66	80 → 37291 [ACK] Seq=1 Ack=100 Win=14480 Len=0 TSval=375524 TSecr=428298
23	52.116711	192.168.50.10	192.168.50.3	HTTP	694	HTTP/1.1 200 OK (text/html)
24	52.116974	192.168.50.3	192.168.50.10	TCP	66	37291 → 80 [ACK] Seq=100 Ack=629 Win=15856 Len=0 TSval=428298 TSecr=375525
> Frame 21: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits)						0000 08 00 27 a3 7c ac 08 00 27 2b f7 02 08 00 45 00E..
Ethernet II, Src: PCSSystemtec_2b:f7:02 (08:00:27:2b:f7:02), Dst: PCSSystemtec_a3:7c:ac (08:00:27:a3:7c:ac)						0010 00 97 15 ce 40 00 40 06 3f 35 c0 a8 32 03 c0 a8 ...@...75.2...
Destination: PCSSystemtec_a3:7c:ac (08:00:27:a3:7c:ac)						0020 32 0a 91 ab 00 50 99 7e 9c c6 1c 5e 2b 4c 80 18 2...@...A...
> Source: PCSSystemtec_2b:f7:02 (08:00:27:2b:f7:02)						0030 03 91 e5 e7 00 00 01 01 08 0a 00 06 89 0a 00 05 ...GET /j ohn,john
Type: IPv4 (0x0800)						0040 ba e4 47 45 54 20 2f 6a 6f 68 6e 2e 6a 6f 68 6e son HTTP /1.1: Us
[Stream index: 4]						0050 73 6f 6e 20 48 54 54 50 2f 31 2e 31 0d 0a 55 73 er-Agent : curl/7
> Internet Protocol Version 4, Src: 192.168.50.3, Dst: 192.168.50.10						0060 65 72 2d 41 67 65 6e 74 3a 20 63 75 72 6c 2f 37 .26.0 .H ost: soc
> Transmission Control Protocol, Src Port: 37291, Dst Port: 80, Seq: 1, Ack: 1, Len: 99						0070 2e 32 36 2e 30 0d 0a 48 6f 73 74 3a 20 73 6f 63 ialsocia lnetwork
> Hypertext Transfer Protocol						0080 69 61 6c 73 6f 63 69 61 6c 6e 65 74 77 6f 72 6b .com .Ac cept: */
						0090 2e 63 6f 6d 0d 0a 41 63 63 65 70 74 3a 20 2a 2f *
						00a0 2a 0d 0a 0d 0a