



**POLYTECHNIQUE
MONTRÉAL**

**UNIVERSITÉ
D'INGÉNIERIE**

INF3405 –Réseaux Informatiques

Hiver 2023

TP No. 2

Groupe [1]

2183376 – Hamza Boukaftane

2113402 – Mehdi El Harami

2244082 – Benzekri Omar

Soumis à :

Mehdi Kadi

09 juin 2023

INTRODUCTION :

Ce travail pratique vise à nous familiariser avec l'utilisation de Wireshark, un outil d'analyse de protocoles réseau. Les réseaux d'aujourd'hui sont de plus en plus complexes, ce qui peut entraîner des dysfonctionnements. L'objectif est de pouvoir localiser et résoudre ces problèmes grâce à l'analyseur de protocoles. En mettant l'interface réseau en mode promiscuous, toutes les trames reçues sont remontées à l'analyseur, permettant ainsi d'identifier les dysfonctionnements. Ce TP nous permettra de comprendre les différents types de paquets, de visualiser l'encapsulation des données et d'analyser les échanges réseau. De plus, il contribuera à développer nos compétences en analyse de problèmes, investigation, utilisation d'outils d'ingénierie et compréhension de l'impact du génie sur la société et l'environnement.

8. Analyse d'une application de traitement d'image :

Nous avons utilisé les adresses IP serveur suivant après avoir fait la commande « ipconfig » au terminal.

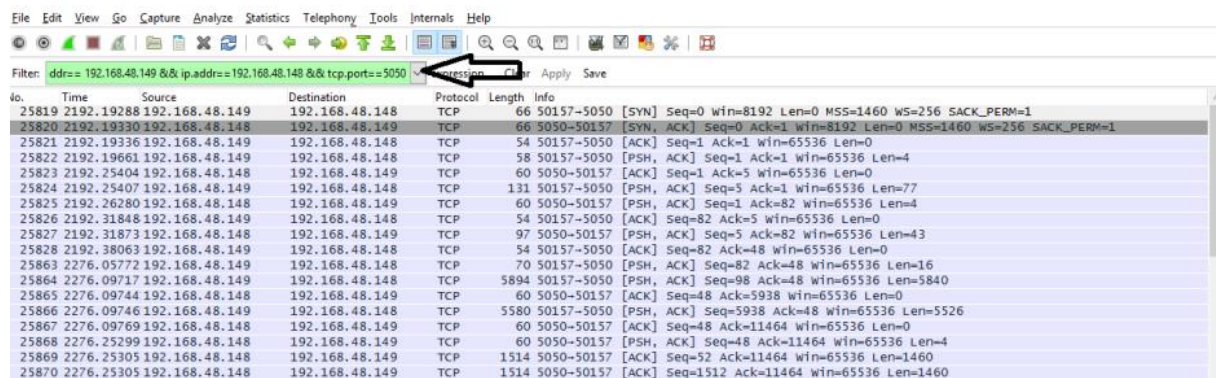
IP serveur: 192.168.48.148

IP client : 192.168.48.149

Numéro de port : 5050

1) Quel filtre appliqueriez-vous afin d'afficher uniquement les échanges entre le client et le serveur ?

On appliquera le filtre suivant : « : ip.addr==192.168.48.148 && ip.addr==192.168.48.149 && tcp.port==5050 » afin d'afficher uniquement les échanges entre le client et le serveur



No.	Time	Source	Destination	Protocol	Length	Info
25819	2192.19288	192.168.48.149	192.168.48.148	TCP	66	50157->5050 [SYN] Seq=0 win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
25820	2192.19330	192.168.48.148	192.168.48.149	TCP	66	5050->50157 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
25821	2192.19336	192.168.48.149	192.168.48.148	TCP	54	50157->5050 [ACK] Seq=1 Ack=1 win=65536 Len=0
25822	2192.19661	192.168.48.149	192.168.48.148	TCP	58	50157->5050 [PSH, ACK] Seq=1 Ack=1 win=65536 Len=4
25823	2192.25404	192.168.48.149	192.168.48.148	TCP	60	5050->50157 [ACK] Seq=1 Ack=5 win=65536 Len=0
25824	2192.25407	192.168.48.149	192.168.48.148	TCP	131	50157->5050 [PSH, ACK] Seq=5 Ack=1 win=65536 Len=77
25825	2192.26280	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [PSH, ACK] Seq=1 Ack=82 win=65536 Len=4
25826	2192.31848	192.168.48.149	192.168.48.148	TCP	54	50157->5050 [ACK] Seq=82 Ack=5 win=65536 Len=0
25827	2192.31873	192.168.48.148	192.168.48.149	TCP	97	5050->50157 [PSH, ACK] Seq=5 Ack=82 win=65536 Len=43
25828	2192.38063	192.168.48.149	192.168.48.148	TCP	54	50157->5050 [ACK] Seq=82 Ack=48 win=65536 Len=0
25863	2276.05772	192.168.48.149	192.168.48.148	TCP	70	50157->5050 [PSH, ACK] Seq=82 Ack=48 win=65536 Len=16
25864	2276.09717	192.168.48.149	192.168.48.148	TCP	5894	50157->5050 [PSH, ACK] Seq=98 Ack=48 win=65536 Len=5840
25865	2276.09744	192.168.48.149	192.168.48.148	TCP	60	5050->50157 [ACK] Seq=48 Ack=5938 win=65536 Len=0
25866	2276.09746	192.168.48.149	192.168.48.148	TCP	5580	50157->5050 [PSH, ACK] Seq=5938 Ack=48 win=65536 Len=5526
25867	2276.09769	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [ACK] Seq=48 Ack=11464 win=65536 Len=0
25868	2276.25299	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [PSH, ACK] Seq=48 Ack=11464 win=65536 Len=4
25869	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=52 Ack=11464 win=65536 Len=1460
25870	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=1512 Ack=11464 win=65536 Len=1460

2) À la lumière de vos observations, dites quel protocole de la couche 4 est utilisé pour la communication entre le client et le serveur.

À la lumière de nos observations, le protocole de la couche 4 utilisé pour la communication client-serveur est le TCP (Transmission Control Protocol). (Voir image prochaine page)

No.	Time	Source	Destination	Protocol	Length	Info
25869	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=52 Ack=11464 win=65536 Len=1460
25870	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=1512 Ack=11464 win=65536 Len=1460
25871	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=2972 Ack=11464 win=65536 Len=1460

No.	Time	Source	Destination	Protocol	Length	Info
25820	2192.19330	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1

No.	Time	Source	Destination	Protocol	Length	Info
0000	00 0c 29 ff 91 35 08 00 45 00					..).S...).S...E.
0010	00 34 5a 15 40 00 80 06					.4j.0...).0...
0020	30 94 c3 ed 13 ba b3 ce					0.....P.....
0030	20 00 e2 a0 00 00 02 04				
0040	04 02					..

3) Combien de paquets et d'octets de données ont été envoyés du client vers le serveur et du serveur vers le client ?

Serveur vers le client :

Le nombre de paquets est de 21.

Taille des données : 17869 octets égale 17.45 ko (Somme des valeurs de la colonne Length de la figure ci-dessous).

No.	Time	Source	Destination	Protocol	Length	Info
25820	2192.19330	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
25823	2192.25404	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [ACK] Seq=1 Ack=5 win=65536 Len=0
25825	2192.26280	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [PSH, ACK] Seq=1 Ack=82 win=65536 Len=4
25827	2192.31873	192.168.48.148	192.168.48.149	TCP	97	5050->50157 [PSH, ACK] Seq=5 Ack=82 win=65536 Len=43
25865	2276.09744	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [ACK] Seq=48 Ack=5938 win=65536 Len=0
25867	2276.09769	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [ACK] Seq=48 Ack=11464 win=65536 Len=0
25868	2276.25299	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [PSH, ACK] Seq=48 Ack=11464 win=65536 Len=4
25869	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=52 Ack=11464 win=65536 Len=1460
25870	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=1512 Ack=11464 win=65536 Len=1460
25871	2276.25305	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=2972 Ack=11464 win=65536 Len=1460
25872	2276.25308	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=4432 Ack=11464 win=65536 Len=1460
25874	2276.25341	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=5892 Ack=11464 win=65536 Len=1460
25875	2276.25342	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=7352 Ack=11464 win=65536 Len=1460
25876	2276.25342	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=8812 Ack=11464 win=65536 Len=1460
25877	2276.25342	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=10272 Ack=11464 win=65536 Len=1460
25878	2276.25342	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=11732 Ack=11464 win=65536 Len=1460
25879	2276.25343	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=13192 Ack=11464 win=65536 Len=1460
25880	2276.25343	192.168.48.148	192.168.48.149	TCP	1514	5050->50157 [ACK] Seq=14652 Ack=11464 win=65536 Len=1460
25881	2276.25343	192.168.48.148	192.168.48.149	TCP	632	5050->50157 [PSH, ACK] Seq=16112 Ack=11464 win=65536 Len=578
25883	2276.25439	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [FIN, ACK] Seq=16690 Ack=11464 win=65536 Len=0
25886	2276.26245	192.168.48.148	192.168.48.149	TCP	60	5050->50157 [ACK] Seq=16691 Ack=11465 win=65536 Len=0

No.	Time	Source	Destination	Protocol	Length	Info
0000	00 0c 29 ff 91 35 08 00 45 00					..).S...).S...E.
0010	00 34 5a 15 40 00 80 06					.4T.0...).0...
0020	30 95 13 ba c3 ed ed 50					0.....P.....
0030	20 00 03 0b 00 00 02 04				
0040	04 02					..

Client vers le serveur :

Le nombre de paquets est de 13.

Taille des données : 12177 octets égale 11.89 ko (Somme des valeurs de la colonne Length même principe que l'image d'en haut).

4) Normalement, le standard IEEE 802.3 limite la taille d'une trame Ethernet à 1518 octets.

Dans votre capture Wireshark, existe-t-il des paquets ayant une taille supérieure à 1518 octets?

Si oui, expliquez pourquoi et comment ce paquet réussit à transiger sur le réseau alors que sa taille est plus grande que celle spécifiée par le standard.

Il existe des paquets ayant une taille supérieure à 1518 octets, car lorsque les paquets sont envoyés du client vers le serveur, la capture s'est faite avant que la carte réseau le fragmente en plus petits paquets puisque la capture se fait du côté du client l'application a envoyé les deux blocs qui dépassent 1518 octets. Voir la figure ci-dessous.

25863	2276.05772	192.168.48.149	192.168.48.148	TCP	70	50157-5050	[PSH, ACK]	Seq=82	Ack=48	Win=65536	Len=16
25864	2276.09717	192.168.48.149	192.168.48.148	TCP	5894	50157-5050	[PSH, ACK]	Seq=98	Ack=48	Win=65536	Len=5840
25866	2276.09746	192.168.48.149	192.168.48.148	TCP	5580	50157-5050	[PSH, ACK]	Seq=5938	Ack=48	Win=65536	Len=5526
25873	2276.25311	192.168.48.149	192.168.48.148	TCP	54	50157-5050	[ACK]	Seq=11464	Ack=5892	Win=65536	Len=0

5) Quel type d'information êtes-vous capables d'extraire de Wireshark en lien avec l'authentification au serveur de traitement d'images ?

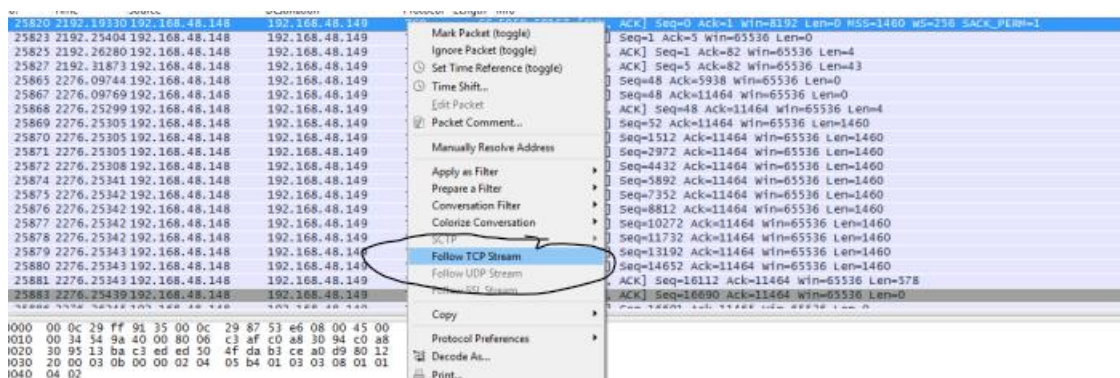
Le nom d'utilisateur : hamza

Le mot de passe : mehdi

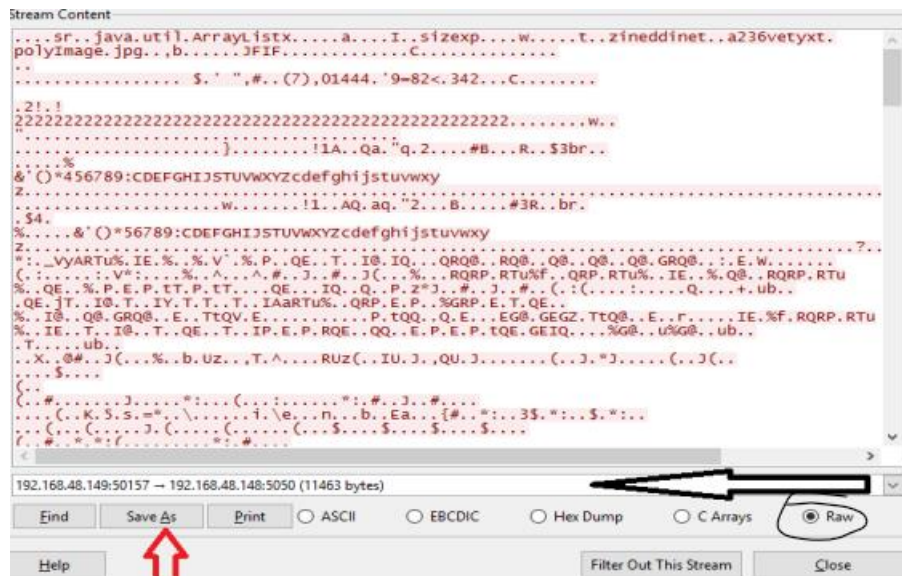
0030	01 00 e2 cc 00 00 73 72	00 13 6a 61 76 61 2e 75sr..java.u
0040	74 69 6c 2e 41 72 72 61	79 4c 69 73 74 78 81 d2	til.ArroyListx..
0050	1d 99 c7 61 9d 03 00 01	49 00 04 73 69 7a 65 78	...a....I..sizex
0060	70 00 00 00 02 77 04 00	00 00 02 74 00 05 68 61	p....w....t.ha
0070	6d 7a 61 74 00 05 6d 65	68 64 69 78	mzat...me hdi

6) Il est possible, avec Wireshark, d'extraire l'image envoyée par le client ou l'image traitée. Donnez les étapes à suivre, incluant des captures d'écran montrant chaque étape permettant l'extraction de l'image envoyée du client vers le serveur. Servez-vous des propriétés du fichier .jpg énoncées plus haut. Indice: utilisez le programme WinHex après avoir sauvegardé le flot de données en format "Raw".

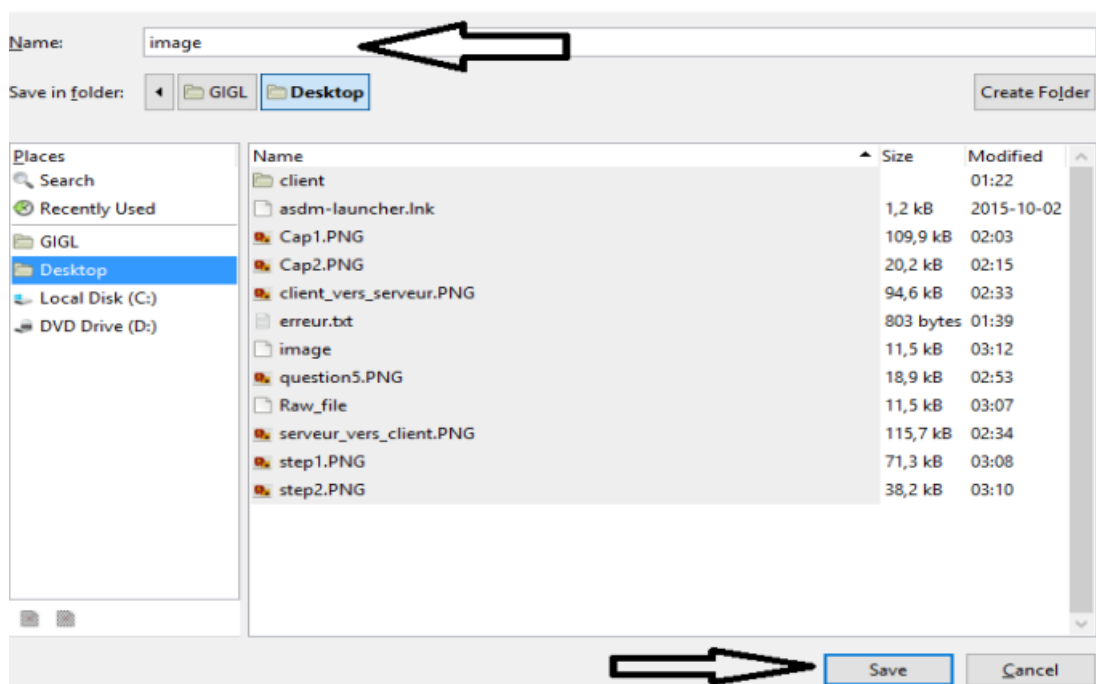
1- Cliquez sur un paquet et sélectionnez Follow TCP stream



- 2- Sélectionnez le flux qui va de la machine client vers la machine serveur et cliquez Save-As, il faut s'assurer de choisir le format Raw.



- 3- Choisissez un nom du fichier de sortie et cliquez sur Save



4- Ouvrez le fichier créé avec WinHex

image																	
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI ASCII
00000000	AC	ED	00	05	73	72	00	13	6A	61	76	61	2E	75	74	69	~i sr java.uti
00000010	6C	2E	41	72	72	61	79	4C	69	73	74	78	81	D2	1D	99	l.ArrayList<> m
00000020	C7	61	9D	03	00	01	49	00	04	73	69	7A	65	78	70	00	Ça I sizexp
00000030	00	00	02	77	04	00	00	00	02	74	00	09	7A	69	6E	65	w t zine
00000040	64	64	69	6E	65	74	00	08	61	32	33	36	76	65	74	79	ddinet a236vety
00000050	78	74	00	0D	70	6F	6C	79	49	6D	61	67	65	2E	6A	70	xt polyImage.jp
00000060	67	00	00	2C	62	FF	D8	FF	E0	00	10	4A	46	49	46	00	g ,byyâ JFIF
00000070	01	02	00	00	01	00	01	00	00	FF	DB	00	43	00	08	06	y0 c
00000080	06	07	06	05	08	07	07	07	09	09	08	0A	0C	14	0D	0C	
00000090	0B	0B	0C	19	12	13	0F	14	1D	1A	1F	1E	1D	1A	1C	1C	
000000A0	20	24	2E	27	20	22	2C	23	1C	1C	28	37	29	2C	30	31	5.' ",# (7),01
000000B0	34	34	34	1F	27	39	3D	38	32	3C	2E	33	34	32	FF	DB	444 '9=2<.342y0
000000C0	00	43	01	09	09	09	0C	0B	0C	18	0D	0D	18	32	21	1C	C 2!
000000D0	21	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	!22222222222222
000000E0	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	22222222222222
000000F0	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	22222222222222
00000100	32	32	32	FF	C0	00	11	08	00	FB	01	77	03	01	22	00	222yA û w "
00000110	02	11	01	03	11	01	FF	C4	00	1F	00	00	01	05	01	01	yA
00000120	01	01	01	01	00	00	00	00	00	00	00	00	01	02	03	04	
00000130	05	06	07	08	09	0A	0B	FF	C4	00	B5	10	00	02	01	03	yA µ
00000140	03	02	04	03	05	05	04	04	00	00	01	7D	01	02	03	00	l
00000150	04	11	05	12	21	31	41	06	13	51	61	07	22	71	14	32	!1A Qa "q 2
00000160	81	91	A1	08	23	42	B1	C1	15	52	D1	F0	24	33	62	72	'! #B±A RÑ053br
00000170	82	09	0A	16	17	18	19	1A	25	26	27	28	29	2A	3A	35	, %s'()*45
00000180	36	37	38	39	3A	43	44	45	46	47	48	49	4A	53	54	55	6789:CD EFGHIJSTU
00000190	56	57	58	59	5A	63	64	65	66	67	68	69	6A	73	74	75	VWXYZcdefghijstu
000001A0	76	77	78	79	7A	83	84	85	86	87	88	89	8A	92	93	94	vwxyzf,;'+*~b\$'""
000001B0	95	96	97	98	99	9A	A2	A3	A4	A5	A6	A7	A8	A9	AA	B2	*~\$&fex!\$'e*~E
000001C0	B3	B4	B5	B6	B7	B8	B9	BA	C2	C3	C4	C5	C6	C7	C8	C9	'µ! . :~AAAA&ÇÈÈ
000001D0	CA	D2	D3	D4	D5	D6	D7	D8	D9	DA	E1	E2	E3	E4	E5	E6	È00000=000aaaa&e
000001E0	E7	E8	E9	EA	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	çèèè000000=000yA
000001F0	00	1F	01	00	03	01	01	01	01	01	01	01	01	01	00	00	
00000200	00	00	00	00	01	02	03	04	05	06	07	08	09	0A	0B	FF	y
00000210	C4	00	B5	11	00	02	01	02	04	04	03	04	07	05	04	04	Å µ
00000220	00	01	02	77	00	01	02	03	11	04	05	21	31	06	12	41	w !l A
00000230	51	07	61	71	13	22	32	81	08	14	42	91	A1	B1	C1	09	Q aq "2 B'±A
00000240	23	33	52	F0	15	62	72	D1	0A	16	24	34	E1	25	F1	17	#3R0 brñ \$4&ñ
00000250	18	19	1A	26	27	28	29	2A	35	36	37	38	39	3A	43	44	&'()*56789:CD

- 5- Sélectionnez les données avant FF D8 FF E0 et après FF D9, s'il y en a et enlevez-les.
(Dans notre cas il y a seulement des données en trop avant FF D8 FF E0)

image																
offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	AC	ED	00	05	73	72	00	13	6A	61	76	61	2E	75	74	69
00000010	6C	2E	41	72	72	61	79	4C	69	73	74	78	81	D2	1D	99
00000020	C7	61	9D	03	00	01	49	00	04	73	69	7A	65	78	70	00
00000030	00	00	02	77	04	00	00	00	02	74	00	09	7A	69	6E	65
00000040	64	64	69	6E	65	74	00	08	61	32	33	36	76	65	74	79
00000050	78	74	00	0D	70	6F	6C	79	49	6D	61	67	65	2E	6A	70
00000060	67	00	00	2C	62	FF	D8	FF	E0	00	10	4A	46	49	46	00
00000070	01	02	00	00	01	00	01	00	00	FF	DB	00	43	00	08	06
00000080	06	07	06	05	08	07	07	07	09	09	08	0A	0C	14	0D	0C
00000090	0B	0B	0C	19	12	13	0F	14	1D	1A	1F	1E	1D	1A	1C	1C

- 6- Cliquez sur Save pour sauvegarder le même fichier avec les modifications. Cliquez ensuite sur Execute et visualisez l'image dans son état de transmission.



- 7) À la suite de toute cette analyse que pouvez-vous conclure quant à la sécurité de l'application de traitement d'images.

L'application de traitement d'images n'est pas tout à fait sécurisée. En effet, les communications transitent sur le réseau et peuvent être interceptées par toutes personnes qui est capable d'enregistrer les paquets échangés. D'où tout les informations, fichiers, document et photos envoyés peuvent être facilement récupérés. C'est pour cela que c'est important d'adopter des mesures de sécurité importantes afin de sécuriser la communication.

10 - Analyse d'une application client-serveur "secrète":

- 1) Quel protocole de la couche transport est utilisé ? Dans le cas de TCP, montrer le tout premier Échange entre le client et le serveur lors de l'initialisation de la connexion, comment ce Nomme cet échange? Dans le cas d'UDP, est-ce que ce même échange à lieu ? Pourquoi ?

Dans le cas de TCP, le premier échange entre le serveur et le client se nomme l'étape de synchronisation ou 'three-way-handshake'. Le but de ce mécanisme est d'établir une connexion fiable

entre deux parties en envoyant et en confirmant une série de paquets de synchronisation (SYN) et d'accusé de réception (ACK).

MODE	PROTOCOLE	PORTS	NOMBRE DE PAQUETS	NOMBRE D'OCTETS DE DONNEES	NOMBRE D'ITERATIONS
1	TCP synchronization/ three-way- handshake	Source = 49865 Destination = 5000	Client→Server = 1 Server→ Client = 0	Client→Server = 1000 Server→ Client = 0	Client→Server = 1
2	TCP synchronization/ three-way- handshake	Source = 49866 Destination = 5000	Client→Server = 25 Server→ Client = 0	Client→Server = 25 x 40 = 1000 Server→ Client = 0	Client→Server = 25
3	UDP	Source = 60072 Destination = 5010	Client→Server = 1 Server→ Client = 0	Client→Server = 3000 Server→ Client = 0	Client→Server = 1
4	UDP	Source = 59410 Destination = 5010	Client→Server = 75 Server→ Client = 0	Client→Server = 75 x 40 = 3000 Server→ Client = 0	Client→Server = 75

MODE 1 :

1	0.00000000	192.168.48.2	192.168.48.148	ICMP	138 Destination unreachable (Host unreachable)
2	1.53055200	192.168.48.2	192.168.48.148	ICMP	138 Destination unreachable (Host unreachable)
3	13.7662750	192.168.48.148	132.207.4.20	SSL	60 Continuation Data
4	13.7662760	132.207.4.20	192.168.48.148	TCP	60 443→49827 [ACK] Seq=1 Ack=2 Win=64240 Len=0
5	14.7201640	192.168.48.148	192.168.48.255	BROWSEF	258 Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain Enum
6	28.5024090	192.168.48.149	192.168.48.148	TCP	66 49865→5000 [ACK] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
7	28.5027450	192.168.48.148	192.168.48.149	TCP	66 5000→49865 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
8	28.5028020	192.168.48.149	192.168.48.148	TCP	54 49865→5000 [ACK] Seq=1 Ack=1 Win=65536 Len=0
9	28.5032320	192.168.48.148	192.168.48.148	TCP	1054 49865→5000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=1000
10	28.5033530	192.168.48.149	192.168.48.148	TCP	54 49865→5000 [FIN, ACK] Seq=1001 Ack=1 Win=65536 Len=0
11	28.5034760	192.168.48.148	192.168.48.149	TCP	60 5000→49865 [ACK] Seq=1 Ack=1002 Win=65536 Len=0
12	28.5037710	192.168.48.148	192.168.48.149	TCP	60 5000→49865 [FIN, ACK] Seq=1 Ack=1002 Win=65536 Len=0
13	28.5038200	192.168.48.149	192.168.48.148	TCP	54 49865→5000 [ACK] Seq=1002 Ack=2 Win=65536 Len=0

```

❏ Frame 9: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits) on interface 0
❏ Ethernet II, Src: Vmware_00:86:5f (00:0c:29:00:86:5f), Dst: Vmware_dc:aa:8a (00:0c:29:dc:aa:8a)
❏ Internet Protocol Version 4, Src: 192.168.48.149 (192.168.48.149), Dst: 192.168.48.148 (192.168.48.148)
❏ Transmission Control Protocol, Src Port: 49865 (49865), Dst Port: 5000 (5000), Seq: 1, Ack: 1, Len: 1000
❏ Data (1000 bytes)

```


MODE 2 :

20	30.2936360	192.168.48.149	192.168.48.148	TCP	66	49866-5000	[SYN]	Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
21	30.2939790	192.168.48.148	192.168.48.149	TCP	66	5000-49866	[SYN, ACK]	Seq=0 Ack=1 win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
22	30.2940750	192.168.48.149	192.168.48.148	TCP	54	49866-5000	[ACK]	Seq=1 Ack=1 win=65536 Len=0
23	30.2944760	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=1 Ack=1 win=65536 Len=40
24	30.2945540	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=41 Ack=1 win=65536 Len=40
25	30.2945890	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=81 Ack=1 win=65536 Len=40
26	30.2946290	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=121 Ack=1 win=65536 Len=40
27	30.2945940	192.168.48.148	192.168.48.149	TCP	60	5000-49866	[ACK]	Seq=1 Ack=81 win=65536 Len=0
28	30.2945330	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=161 Ack=1 win=65536 Len=40
29	30.2947960	192.168.48.148	192.168.48.149	TCP	60	5000-49866	[ACK]	Seq=1 Ack=161 win=65536 Len=0
30	30.2948200	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=201 Ack=1 win=65536 Len=40
31	30.2948970	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=241 Ack=1 win=65536 Len=40
32	30.2949400	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=281 Ack=1 win=65536 Len=40
33	30.2950040	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=321 Ack=1 win=65536 Len=40
34	30.2950400	192.168.48.148	192.168.48.149	TCP	60	5000-49866	[ACK]	Seq=1 Ack=241 win=65536 Len=0
35	30.2950440	192.168.48.149	192.168.48.148	TCP	94	49866-5000	[PSH, ACK]	Seq=361 Ack=1 win=65536 Len=40

Frame 28: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface 0
Ethernet II, Src: Vmware_00:86:5f (00:0c:29:00:86:5f), Dst: Vmware_dc:aa:8a (00:0c:29:dc:aa:8a)
Internet Protocol Version 4, Src: 192.168.48.149 (192.168.48.149), Dst: 192.168.48.148 (192.168.48.148)
Transmission Control Protocol, Src Port: 49866 (49866), Dst Port: 5000 (5000), Seq: 161, Ack: 1, Len: 40
Data (40 bytes)

MODE 3 :

1	0.00000000	192.168.48.149	192.168.48.148	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=0, ID=3e7f)
2	0.00000700	192.168.48.149	192.168.48.148	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=1480, ID=3e7f) [Reassembled in #3]
3	0.00000900	192.168.48.149	192.168.48.148	UDP	82	Source port: 60072 Destination port: 5010
4	0.00038100	192.168.48.148	192.168.48.149	ICMP	590	Destination unreachable (Port unreachable)
5	4.53431600	Vmware_dc:aa:8a	Vmware_00:86:5f	ARP	60	who has 192.168.48.149? Tell 192.168.48.148
6	4.53433100	Vmware_00:86:5f	Vmware_dc:aa:8a	ARP	42	192.168.48.149 is at 00:0c:29:00:86:5f
7	4.73399400	192.168.48.1	192.168.48.255	BROWSE	243	Host Announcement L4708-22, workstation, Server, NT workstation
8	4.83475000	Vmware_00:86:5f	Vmware_dc:aa:8a	ARP	42	who has 192.168.48.148? Tell 192.168.48.149
9	4.83500000	Vmware_dc:aa:8a	Vmware_00:86:5f	ARP	60	192.168.48.148 is at 00:0c:29:dc:aa:8a
10	13.66934200	192.168.48.148	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
11	14.68517400	192.168.48.148	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
12	14.93105400	192.168.48.148	132.207.4.20	SSL	60	Continuation data
13	14.93105500	132.207.4.20	192.168.48.148	TCP	60	443-49827 [ACK] Seq=1 Ack=2 Win=64240 Len=0
14	15.69251000	192.168.48.148	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
15	16.69308600	192.168.48.148	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
16	36.61206800	192.168.48.148	192.168.48.2	NBNS	110	Refresh NB <01><02>_MSBROWSE__<02><01>
17	38.22397000	192.168.48.148	192.168.48.2	NBNS	110	Refresh NB <01><02>_MSBROWSE__<02><01>
18	39.84922000	192.168.48.148	192.168.48.2	NBNS	110	Refresh NB <01><02>_MSBROWSE__<02><01>

Frame 3: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface 0
Ethernet II, Src: Vmware_00:86:5f (00:0c:29:00:86:5f), Dst: Vmware_dc:aa:8a (00:0c:29:dc:aa:8a)
Internet Protocol Version 4, Src: 192.168.48.149 (192.168.48.149), Dst: 192.168.48.148 (192.168.48.148)
User Datagram Protocol, Src Port: 60072 (60072), Dst Port: 5010 (5010)
Data (3000 bytes)

MODE 4 :

3	3.81968200	192.168.48.149	65.55.252.93	TCP	66	[TCP Retransmission] 49874-443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
6	5.14762700	13.107.21.200	192.168.48.149	TCP	60	443-49870 [RST, ACK] Seq=1 Ack=1 win=64240 Len=0
7	8.83458600	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
8	8.83465800	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
9	8.83474900	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
10	8.83481200	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
11	8.83486300	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
12	8.83490300	192.168.48.148	192.168.48.149	ICMP	110	Destination unreachable (Port unreachable)
13	8.83490500	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
14	8.83496400	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
15	8.83503900	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
16	8.83506400	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
17	8.83508600	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
18	8.83514200	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
19	8.83518700	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
20	8.83528200	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
21	8.83532700	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010
22	8.83537700	192.168.48.149	192.168.48.148	UDP	82	Source port: 59410 Destination port: 5010

Frame 7: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface 0
Ethernet II, Src: Vmware_00:86:5f (00:0c:29:00:86:5f), Dst: Vmware_dc:aa:8a (00:0c:29:dc:aa:8a)
Internet Protocol Version 4, Src: 192.168.48.149 (192.168.48.149), Dst: 192.168.48.148 (192.168.48.148)
User Datagram Protocol, Src Port: 59410 (59410), Dst Port: 5010 (5010)
Data (40 bytes)

C) Analyse des performances et protocole TCP :

- 1) **Comparez la performance des envois de données pour le mode 1 et le mode 2. Qu'est-ce qui diffère entre ces deux modes? Lequel est le plus performant selon vous et pourquoi ?**

Le mode 1 transfère ces données avec un seul paquet avec une faible surcharge de connexion et de gestion des paquets la taille des paquets est 1000 octets ce qui réduit les performances si le volume de données dépasse cette limite. Tandis que le mode 2 transfère une plus grande quantité, mais en 25 paquets avec chacun 40 octets de données. On ne nécessite pas une nouvelle connexion pour chaque paquet, mais cela peut entraîner une surcharge de gestion de paquets et donc réduire la performance. On déduit donc que le mode 1 est plus performant que le mode 2, car il encapsule qu'une seule fois l'entête du TCP contrairement au mode 2 qui l'encapsule 25 fois.

- 2) **Comparer la performance des envois de données pour le mode 3 et le mode 4. Qu'est-ce qui diffère entre ces deux modes? Lequel est le plus performant selon vous et pourquoi**

Le mode 4 transfère une plus grande quantité de données que le mode 3. En effet, le mode 4 utilise 75 paquets avec chacun des paquets 40 octets sans une nouvelle connexion pour chaque paquet. Tandis que le mode 3 a une faible surcharge au niveau des gestions des paquets, mais le transfert de données est limité à 3000 octets. Le mode 4 est plus performant que le mode 3 dans le cas de l'UDP, car en cas de perte de paquets, on est toujours apte à récupérer une partie des données tandis qu'au mode 3 on perdra la totalité du paquet.

- 3) **Discutez de la fiabilité de chaque mode. Selon vous, quel(s) mode(s) est le plus fiable ?**

Mode 1 : assure la fiabilité des transferts de données en revoyant tout paquet perdu ou corrompu grâce à l'utilisation du protocole TCP.

Mode 2 : utilise le protocole TCP, les paquets sont de plus petites tailles que celle du mode 1 impliquant une perte de performance lorsqu'on devra transférer de grandes quantités de données.

Mode 3 : contrairement au mode 1, n'assure pas la fiabilité de transfert, car il n'y a pas de système de vérification de réception. En effet, le mode 3 utilise le protocole UDP expliquant la perte de paquets.

Mode 4 : utilise le protocole UDP, les paquets sont plus petites que celle du mode 3 réduisant la perte de données s'il y a perte de paquets. Cependant, il important de noter que la fiabilité de transfert de donnée n'est toujours pas assurée.

Enfin, le mode 1 est le plus fiable. En effet, malgré sa perte de performance lors de surcharge, il demeure très fiable pour le transfert de données.

4) Pour les modes secrets utilisant le protocole TCP, vous avez certainement remarqué à la fin de la communication un échange FIN, ACK. Expliquez en quoi consiste cet échange.

Lorsqu'un des acteurs, client ou serveur, décide de fermer la connexion TCP, il transmet un paquet FIN signifiant finish à l'autre acteur de la connexion afin d'indiquer la fin de l'envoi de données. Le récepteur du paquet FIN répond alors en transmettant un paquet ACK signifiant acknowledgment afin de confirmer la réception du paquet FIN. De cette façon, la communication est terminée de manière structurée tout en libérant adéquatement les ressources utilisées.