Travaux pratiques 4.5.2 : Protocoles TCP et UDP de la couche transport TCP/IP

Schéma de topologie

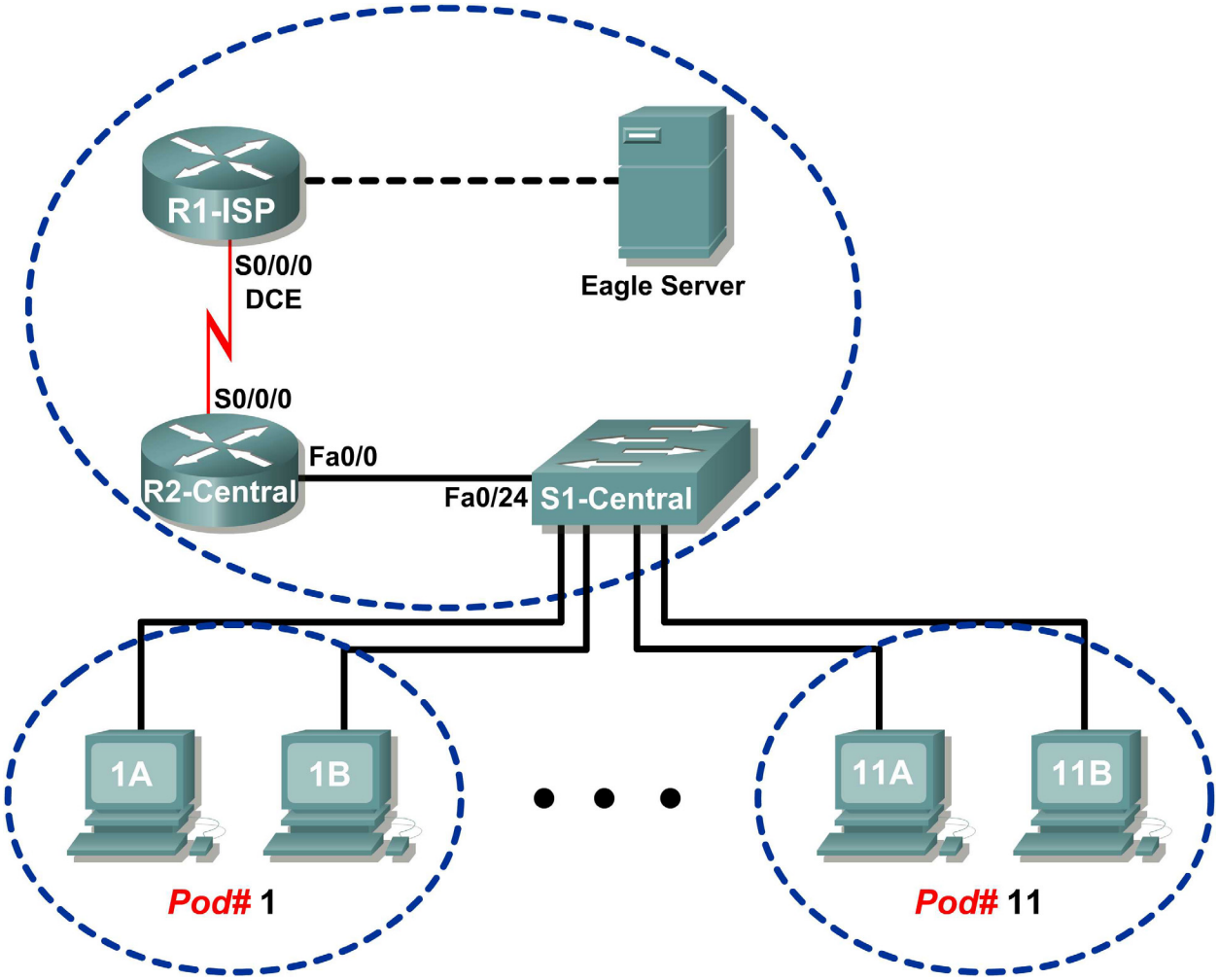

Table d'adressage

Périphérique	Interface	Adresse IP	Masque de sous-réseau	Passerelle par défaut
R1-ISP	S0/0/0	10.10.10.6	255.255.255.252	S/O
	Fa0/0	192.168.254.253	255.255.255.0	S/O
R2-Central	S0/0/0	10.10.10.5	255.255.255.252	S/O
	Fa0/0	172.16.255.254	255.255.0.0	S/O
Eagle Server	S/O	192.168.254.254	255.255.255.0	192.168.254.253
	S/O	172.31.24.254	255.255.255.0	S/O
host*Pod*#A	S/O	172.16.*Pod*#.1	255.255.0.0	172.16.255.254
host*Pod*#B	S/O	172.16.*Pod*#.2	255.255.0.0	172.16.255.254
S1-Central	S/O	172.16.254.1	255.255.0.0	172.16.255.254

Objectifs pédagogiques

- Identifier les champs d'en-tête TCP ainsi que les opérations à l'aide de la capture de session FTP de Wireshark.
- Identifier les champs d'en-tête UDP ainsi que les opérations à l'aide de la capture de session TFTP de Wireshark.

Contexte

Les deux protocoles dans la couche transport du modèle TCP/IP sont TCP (Transmission Control Protocol), défini dans la RFC 761 de janvier 1980, et UDP (User Datagram Protocol), défini dans la RFC 768 d'août 1980. Ces deux protocoles prennent en charge la communication de protocoles de couche supérieure. Par exemple, TCP permet d'offrir la prise en charge de la couche transport pour les protocoles HTTP et FTP, entre autres. Quant à UDP, il fournit cette prise en charge pour les services de noms de domaines (DNS) et le protocole TFTP (Trivial File Transfer Protocol), entre autres.

La capacité à comprendre les éléments des en-têtes TCP et UDP ainsi que les opérations représentent une compétence cruciale pour les ingénieurs réseau.

Scénario

À l'aide de la capture Wireshark, analysez les champs d'en-têtes des protocoles TCP et UDP pour les transferts de fichiers entre l'ordinateur hôte et Eagle Server. Si vous n'avez pas effectué le téléchargement de Wireshark sur l'ordinateur hôte pod, utilisez l'adresse URL `ftp://eagle-server.example.com/pub/eagle labs/eagle1/chapter4/`, fichier `wireshark-setup-0.99.4.exe`.

Les utilitaires de ligne de commande Windows `ftp` et `tftp` servent à la connexion à Eagle Server et au téléchargement des fichiers.

Tâche 1 : identification des champs d'en-tête TCP ainsi que les opérations à l'aide de la capture de session FTP de Wireshark.

Étape 1 : capture d'une session FTP.

Les sessions TCP sont parfaitement contrôlées et gérées par les informations échangées dans les champs d'en-tête TCP. Dans cette tâche, une session FTP est effectuée dans Eagle Server. Ensuite, la capture de session est analysée. Les ordinateurs Windows utilisent le client FTP, **`ftp`**, pour la connexion au serveur FTP. Une fenêtre de ligne de commande démarre la session FTP, et le fichier de configuration du texte pour S1-central est téléchargé depuis Eagle Server, `/pub/eagle_labs/eagle1/chapter4/s1-central`, vers l'ordinateur hôte.

Ouvrez une fenêtre de ligne de commande en cliquant sur Démarrer | Exécuter, tapez cmd, puis appuyez sur OK.

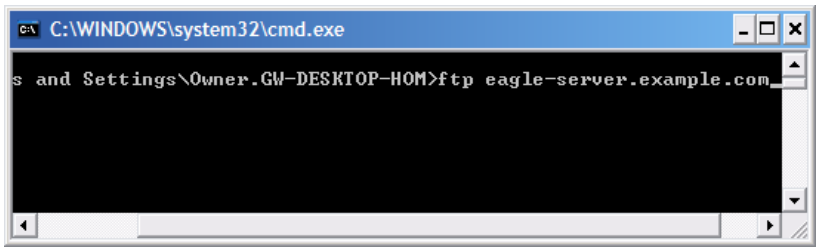

Figure 1. Fenêtre de ligne de commande

Une fenêtre semblable à la figure 1 doit s'afficher.

Démarrez une capture Wireshark sur l'interface dont l'adresse IP est `172.16.`*`Pod#`*`.[1-2].`

Démarrez une connexion FTP à Eagle Server. Tapez la commande :

```
> ftp eagle-server.example.com
```

À l'invite d'un ID utilisateur, tapez **`anonyme`**. Lorsque le système vous demande un mot de passe, appuyez sur **`<ENTRÉE>`**.

Allez au répertoire FTP `/pub/eagle_labs/eagle1/chapter4/` :

```
ftp> cd /pub/eagle_labs/eagle1/chapter4/
```

Téléchargez le fichier `s1-central` :

```
ftp> get s1-central
```

Ensuite, fermez les sessions FTP dans chaque fenêtre de ligne de commande avec la commande **`quit`** :

```
ftp> quit
```

Fermez la fenêtre de ligne de commande avec la commande **`exit`** :

```
> exit
```

Arrêtez la capture Wireshark.

Étape 2 : analyse des champs TCP.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	172.16.1.1	192.168.254.254	TCP	1052 > ftp [SYN] Seq=0 Len=0 MSS=1460
2	0.000568	192.168.254.254	172.16.1.1	TCP	ftp > 1052 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
3	0.000610	172.16.1.1	192.168.254.254	TCP	1052 > ftp [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	0.004818	192.168.254.254	172.16.1.1	FTP	Response: 220 Welcome to the eagle-server FTP service.
5	0.115430	172.16.1.1	192.168.254.254	TCP	1052 > ftp [ACK] Seq=1 Ack=47 Win=64194 Len=0
6	8.223541	172.16.1.1	192.168.254.254	FTP	Request: USER anonymous
7	8.224089	192.168.254.254	172.16.1.1	TCP	ftp > 1052 [ACK] Seq=47 Ack=17 Win=5840 Len=0
8	8.224126	192.168.254.254	172.16.1.1	FTP	Response: 331 Please specify the password.
9	8.327214	172.16.1.1	192.168.254.254	TCP	1052 > ftp [ACK] Seq=17 Ack=81 Win=64160 Len=0
10	9.517629	172.16.1.1	192.168.254.254	FTP	Request: PASS
11	9.519135	192.168.254.254	172.16.1.1	FTP	Response: 230 Login successful.
12	9.629097	172.16.1.1	192.168.254.254	TCP	1052 > ftp [ACK] Seq=24 Ack=104 Win=64137 Len=0
13	32.365752	172.16.1.1	192.168.254.254	FTP	Request: CWD /pub/eagle_labs/eagle1/chapter4
14	32.366375	192.168.254.254	172.16.1.1	FTP	Response: 250 Directory successfully changed.
15	32.376653	172.16.1.1	192.168.254.254	FTP	Request: PORT 172,16,1,1,4,33
16	32.377165	192.168.254.254	172.16.1.1	FTP	Response: 200 PORT command successful. Consider using PASV.
17	32.381726	172.16.1.1	192.168.254.254	FTP	Request: RETR s1-central
18	32.382337	192.168.254.254	172.16.1.1	TCP	ftp-data > 1057 [SYN] Seq=0 Len=0 MSS=1460 TSV=4755496 TSER=0 WS=2
19	32.382398	172.16.1.1	192.168.254.254	TCP	1057 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 WS=0 TSV=0 TSER=0
20	32.382777	192.168.254.254	172.16.1.1	TCP	ftp-data > 1057 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSV=4755496 TSER=0
21	32.382891	192.168.254.254	172.16.1.1	FTP	Response: 150 Opening BINARY mode data connection for s1-central (3100 bytes).
22	32.383528	192.168.254.254	172.16.1.1	FTP-DATA	FTP Data: 1448 bytes
23	32.383589	192.168.254.254	172.16.1.1	FTP-DATA	FTP Data: 1448 bytes
24	32.383631	172.16.1.1	192.168.254.254	TCP	1057 > ftp-data [ACK] Seq=1 Ack=2897 Win=64240 Len=0 TSV=36854 TSER=4755496
25	32.383736	192.168.254.254	172.16.1.1	FTP-DATA	FTP Data: 204 bytes
26	32.383753	192.168.254.254	172.16.1.1	FTP	Response: 226 File send OK.
27	32.383773	172.16.1.1	192.168.254.254	TCP	1052 > ftp [ACK] Seq=100 Ack=281 Win=63960 Len=0
28	32.383779	192.168.254.254	172.16.1.1	TCP	ftp-data > 1057 [FIN, ACK] Seq=3101 Ack=1 Win=5840 Len=0 TSV=4755496 TSER=0
29	32.383809	172.16.1.1	192.168.254.254	TCP	1057 > ftp-data [ACK] Seq=1 Ack=3102 Win=64036 Len=0 TSV=36854 TSER=4755496
30	32.389457	172.16.1.1	192.168.254.254	TCP	1057 > ftp-data [FIN, ACK] Seq=1 Ack=3102 Win=64036 Len=0 TSV=36854 TSER=4755496
31	32.389845	192.168.254.254	172.16.1.1	TCP	ftp-data > 1057 [ACK] Seq=3102 Ack=2 Win=5840 Len=0 TSV=4755503 TSER=36854
32	34.438952	172.16.1.1	192.168.254.254	FTP	Request: QUIT
33	34.439532	192.168.254.254	172.16.1.1	FTP	Response: 221 Goodbye.
34	34.439893	192.168.254.254	172.16.1.1	TCP	ftp > 1052 [FIN, ACK] Seq=295 Ack=106 Win=5840 Len=0
35	34.439934	172.16.1.1	192.168.254.254	TCP	1052 > ftp [ACK] Seq=106 Ack=296 Win=63946 Len=0
36	34.442705	172.16.1.1	192.168.254.254	TCP	1052 > ftp [FIN, ACK] Seq=106 Ack=296 Win=63946 Len=0
37	34.443144	192.168.254.254	172.16.1.1	TCP	ftp > 1052 [ACK] Seq=296 Ack=107 Win=5840 Len=0

Figure 2. Capture FTP

Basculez vers les fenêtres de capture Wireshark. La fenêtre supérieure contient les informations récapitulatives pour chaque enregistrement capturé. La capture par le participant doit être semblable à celle illustrée à la figure 2. Avant d'approfondir le concept de paquet TCP, une explication des informations récapitulatives s'impose. Lorsque le client FTP est connecté au serveur FTP, le protocole TCP de la couche transport a créé une session fiable. TCP est couramment utilisé au cours d'une session pour contrôler la transmission et l'arrivée des datagrammes ainsi que pour gérer la taille des fenêtres. Pour chaque échange de données entre le client FTP et le serveur FTP, une session TCP est créée. Au terme du transfert de données, la session TCP est fermée. Ainsi, une fois la session FTP terminée, TCP exécute un arrêt et une déconnexion normalement.

```
Transmission Control Protocol, Src Port: 1052 (1052), Dst Port: ftp (21), Seq: 0, Len: 0
    Source port: 1052 (1052)
    Destination port: ftp (21)
    Sequence number: 0    (relative sequence number)
    Header length: 28 bytes
  Flags: 0x02 (SYN)
      0... .... = Congestion Window Reduced (CWR): Not set
      .0.. .... = ECN-Echo: Not set
      ..0. .... = Urgent: Not set
      ...0 .... = Acknowledgment: Not set
      .... 0... = Push: Not set
      .... .0.. = Reset: Not set
      .... ..1. = Syn: Set
      .... ...0 = Fin: Not set
    Window size: 64240
    Checksum: 0xb965 [correct]
  Options: (8 bytes)
      Maximum segment size: 1460 bytes
      NOP
      NOP
      SACK permitted
```

Figure 3. Capture Wireshark d'un datagramme TCP

Dans Wireshark, les informations TCP détaillées sont disponibles dans la fenêtre du milieu. Sélectionnez le premier datagramme TCP à partir de l'ordinateur hôte, et déplacez le pointeur de la souris vers la fenêtre du milieu. Il peut s'avérer nécessaire de modifier la fenêtre du milieu et de développer l'enregistrement TCP en cliquant sur la zone de développement du protocole. Le datagramme TCP développé doit être semblable à la figure 3.

Quelle est la méthode d'identification du premier datagramme dans une session TCP ?

_____

_____

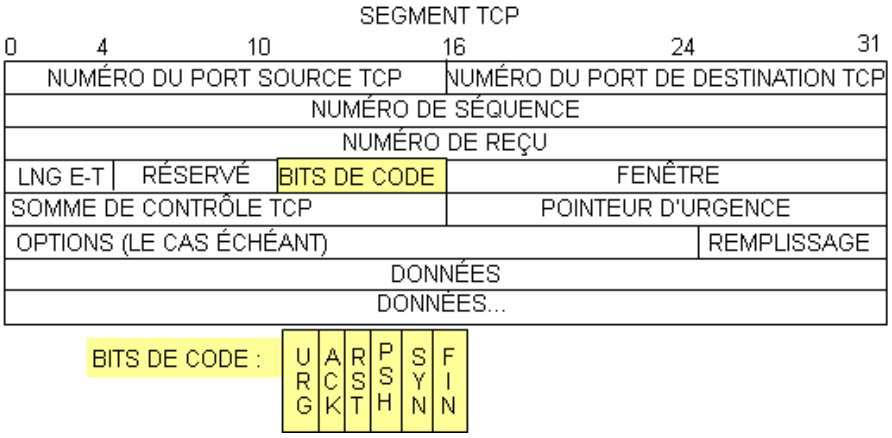

Figure 4. Champs de paquet TCP

Reportez-vous à la figure 4, un schéma de datagramme TCP. Une explication de chaque champ est disponible pour rafraîchir la mémoire du participant :

- **Le numéro de port source TCP** appartient à l'hôte de session TCP qui a ouvert une connexion. Il s'agit généralement d'une valeur aléatoire supérieure à 1023.
- **Le numéro de port de destination** permet d'identifier le protocole de couche supérieure ou l'application sur le site distant. Les valeurs dans la plage 0–1023 représentent les « ports bien connus » et sont associées aux services et aux applications standard (selon la description dans la RFC 1700, comme Telnet, FTP (File Transfer Protocol), HTTP (HyperText Transfer Protocol), etc.). La combinaison de quatre champs (Adresse IP source, Port source, Adresse IP de destination, Port de destination) identifie de façon unique la session à l'émetteur et au récepteur.
- **Le numéro d'ordre** indique le numéro du dernier octet dans un segment.
- **Le numéro de reçu** indique l'octet suivant prévu par le récepteur.
- **les Bits de code** ont une signification spécifique dans la gestion des sessions et dans le traitement des segments. Valeurs intéressantes :
    - ACK (reçu d'un segment) ;
    - SYN (Synchronize, uniquement défini lorsqu'une nouvelle session TCP est négociée au cours de la connexion en trois étapes) ;
    - FIN (Finish, requête pour fermer la session TCP) ;
- **La taille de la fenêtre** est la valeur de la fenêtre glissante : le nombre d'octets qui peuvent être envoyés avant d'attendre le reçu.
- **Le pointeur d'urgence** n'est utilisé qu'avec un indicateur URG (Urgent) : lorsque l'émetteur doit envoyer des données urgentes au récepteur.
- **Options** : la seule option actuellement définie est la taille de segment TCP maximale (valeur facultative).

À l'aide de la capture Wireshark du démarrage de la première session TCP (bit SYNC défini sur 1), renseignez les informations concernant l'en-tête TCP :

De l'ordinateur hôte pod vers Eagle Server (seul le bit SYN est défini sur 1) :

Adresse IP source : 172.16.___.___	
Adresse IP de destination : _____	
Numéro du port source : _____	
Numéro du port de destination : _____	
Numéro d'ordre : _____	
Numéro de reçu : _____	
Longueur de l'en-tête : _____	
Taille de fenêtre : _____	

D'Eagle Server vers l'ordinateur hôte pod (seuls les bits SYN et ACK sont définis sur 1) :

Adresse IP source : _____	
Adresse IP de destination : 172.16.___.___	
Numéro du port source : _____	
Numéro du port de destination :_____	
Numéro d'ordre : _____	
Numéro de reçu : _____	
Longueur de l'en-tête : _____	
Taille de fenêtre : _____	

De l'ordinateur hôte pod vers le Eagle Server (seul le bit ACK est défini sur 1) :

Adresse IP source : 172.16.___.___	
Adresse IP de destination : _____	
Numéro du port source : _____	
Numéro du port de destination : _____	
Numéro d'ordre : _____	
Numéro de reçu : _____	
Longueur de l'en-tête : _____	
Taille de fenêtre : _____	

Sans tenir compte de la session IP démarrée lors d'un transfert de données, combien d'autres datagrammes TCP contenaient un bit SYN ?

_____

_____

Les pirates informatiques profitent de la connexion en trois étapes en amorçant une connexion « semi-ouverte ». Dans cette séquence, la session TCP d'ouverture envoie un datagramme TCP avec le bit SYN défini. En outre, le récepteur envoie un datagramme TCP associé avec les bits SYN ACK définis. Un bit ACK final n'est jamais envoyé pour terminer la connexion TCP. À la place, une nouvelle connexion TCP est démarrée de manière semi-ouverte. Avec un nombre suffisant de sessions TCP dans l'état semi-ouvert, l'ordinateur récepteur risque d'épuiser les ressources et de tomber en panne. Cela pourrait entraîner une perte de services réseau ou endommager le système d'exploitation. Dans les deux cas, le pirate informatique a gagné. Le service réseau a été arrêté sur le récepteur. Ceci est un exemple d'attaque par déni de service.

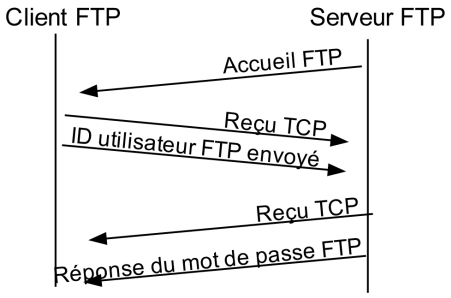

Figure 5. Gestion des sessions TCP

Le client FTP et le serveur communiquent entre eux sans tenir compte du contrôle et de la gestion de la session par TCP. Lorsque le serveur FTP envoie une réponse : 220 au client FTP, la session TCP sur le client FTP envoie un reçu à la session TCP sur Eagle Server. Cette séquence est illustrée à la figure 5, et est visible dans la capture Wireshark.

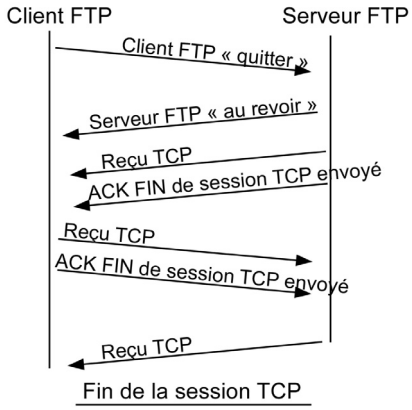

Figure 6. Fin normale de la session TCP

Une fois la session FTP terminée, le client FTP envoie une commande pour « `quitter` ». Le serveur FTP accuse réception de la fin de la session FTP avec un message `Response :221 Goodbye`. À ce stade, la session TCP du serveur FTP envoie un datagramme TCP au client FTP, et annonce ainsi la fin de la session TCP. La session TCP du client FTP accuse réception du datagramme de fin, puis envoie la fin de sa propre session TCP. Lorsque l'émetteur de la fin de la session TCP, le serveur FTP, reçoit une fin en double, un datagramme ACK est envoyé pour accuser réception de la fin et la session TCP est fermée. Cette séquence est illustrée à la figure 6, et est visible dans la capture Wireshark.

Sans fin normale, comme dans le cas d'une connexion rompue, les sessions TCP attendent un certain délai avant la fermeture. Le délai d'attente varie, mais il est généralement de 5 minutes.

Tâche 2 : identification des champs d'en-tête UDP ainsi que les opérations à l'aide de la capture de session TFTP de Wireshark.

Étape 1 : capture d'une session TFTP.

Suivant la procédure dans la tâche 1 ci-dessus, ouvrez une fenêtre de ligne de commande. La commande TFTP possède une syntaxe différente de FTP. Par exemple, l'authentification n'est pas disponible. En outre, il n'existe que deux commandes, **`get`**, pour récupérer un fichier et **`put`**, pour envoyer un fichier.


```
>tftp -help

Transfère des fichiers vers/depuis un ordinateur distant exécutant le
service TFTP.

TFTP [-i] host [GET | PUT] source [destination]

        -i          Indique le mode de transfert d'image binaire
                    (également nommé octet). En mode d'image binaire, le
                    fichier est déplacé littéralement, octet par octet.
                    Utilisez ce mode lors du transfert des fichiers
                    binaires.
        host        Spécifie l'hôte local ou distant.
        GET         transfère la destination du fichier sur l'hôte distant
                    vers la source du fichier sur l'hôte local.
        PUT         Transfère la source du fichier sur l'hôte local vers
                    la destination du fichier sur l'hôte distant.
        source      Indique le fichier à transférer.
  destination       Indique où transférer le fichier.
```

Tableau 1. Syntaxe TFTP pour un client TFTP de Windows

Le tableau 1 contient la syntaxe du client TFTP de Windows. Le serveur TFTP possède son propre répertoire sur Eagle Server, `/tftpboot`, qui est différent de la structure de répertoires prise en charge par le serveur FTP. L'authentification n'est pas prise en charge.

Démarrez une capture Wireshark, puis téléchargez le fichier de configuration `s1-central` à partir d'Eagle Server avec le client TFTP de Windows. La commande et la syntaxe pour effectuer cette opération sont indiquées ci-dessous :

```
>tftp eagle-server.example.com get s1-central
```

Étape 2 : analyse des champs UDP.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	172.16.1.1	192.168.254.254	TFTP	Read Request, File: s1-central, Transfer type: netascii
2	0.003171	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 1
3	0.003314	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 1
4	0.003962	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 2
5	0.004021	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 2
6	0.004615	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 3
7	0.004673	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 3
8	0.005274	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 4
9	0.005332	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 4
10	0.005930	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 5
11	0.005989	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 5
12	0.006588	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 6
13	0.006644	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 6
14	0.007078	192.168.254.254	172.16.1.1	TFTP	Data Packet, Block: 7 (last)
15	0.007131	172.16.1.1	192.168.254.254	TFTP	Acknowledgement, Block: 7

Figure 7. Résumé d'une capture d'une session UDP

Basculez vers les fenêtres de capture Wireshark. La capture par le participant doit être semblable à celle illustrée à la figure 7. Un transfert TFTP permet d'analyser les opérations UDP de la couche transport.

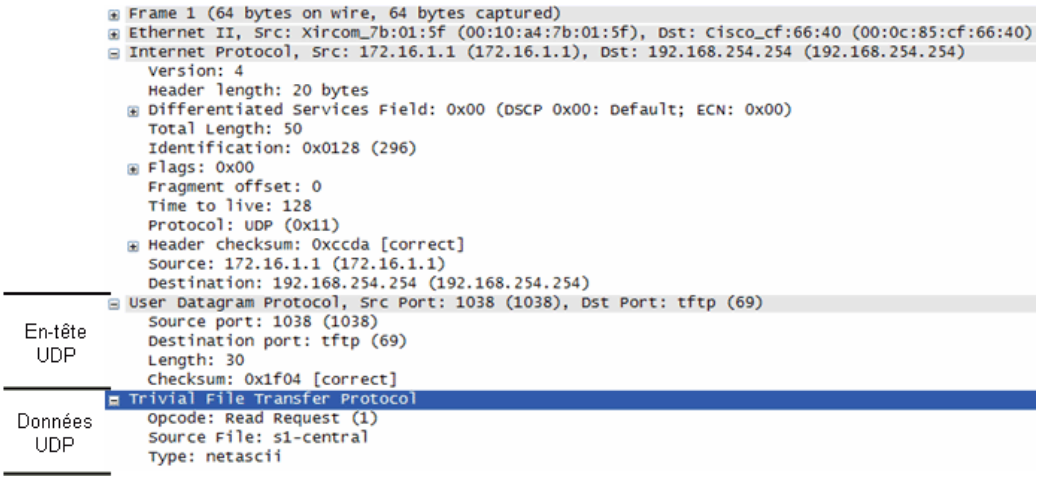

Figure 8. Capture Wireshark d'un datagramme UDP

Dans Wireshark, les informations UDP détaillées sont disponibles dans la fenêtre du milieu. Sélectionnez le premier datagramme UDP à partir de l'ordinateur hôte, et déplacez le pointeur de la souris vers la fenêtre du milieu. Il peut s'avérer nécessaire de modifier la fenêtre du milieu et de développer l'enregistrement UDP en cliquant sur la zone de développement du protocole. Le datagramme UDP développé doit être semblable à la figure 8.

SEGMENT UDP

0	16	31
PORT SOURCE UDP	PORT DE DESTINATION UDP	
LONGUEUR DU MESSAGE UDP	SOMME DE CONTRÔLE UDP	
DONNÉES		
DONNÉES...		

Figure 9. Format UDP

Reportez-vous à la figure 9, un schéma de datagramme UDP. Les informations d'en-tête sont peu nombreuses par rapport au datagramme TCP. Des similitudes existent cependant. Chaque datagramme UDP est identifié par les ports source et de destination UDP.

À l'aide de la capture Wireshark du premier datagramme IDP, renseignez les informations concernant l'en-tête UDP. La somme de contrôle est une valeur hexadécimale (base 16), identifiée par le code 0x précédent :

Adresse IP source : 172.16.___.___	
Adresse IP de destination : _____	
Numéro du port source : _____	
Numéro du port de destination : _____	
Longueur du message UDP : _____	
Somme de contrôle UDP : _____	

De quelle manière UDP vérifie-t-il l'intégrité du datagramme ?

_____

_____

Examinez le premier paquet retourné par Eagle Server. Renseignez les informations sur l'en-tête UDP :

Adresse IP source :	
Adresse IP de destination : 172.16.___.___	
Numéro du port source : _____	
Numéro du port de destination : _____	
Longueur du message UDP : _____	
Somme de contrôle UDP : 0x _____	

Remarque : le datagramme UDP de retour possède un port de source UDP différent. Toutefois, ce dernier sert au transfert TFTP restant. Comme la connexion n'est pas fiable, seul le port source d'origine utilisé pour commencer la session TFTP sert à gérer le transfert TFTP.

Tâche 5 : remarques générales.

Ces travaux pratiques ont permis aux participants d'analyser les opérations des protocoles TCP et UDP à partir de sessions FTP et TFTP capturées. La gestion de la communication par TCP est très différente de celle par UDP. Toutefois, la fiabilité et la transmission garantie nécessitent un contrôle supplémentaire sur le canal de communication. UDP comporte moins de surcharge et de contrôle, et le protocole de couche supérieure doit fournir un certain type de contrôle des reçus. Les deux protocoles, cependant, transportent des données entre les clients et les serveurs à l'aide des protocoles de la couche application. En outre, ils sont applicables au protocole de couche supérieure que chacun prend en charge.

Tâche 6 : confirmation.

Comme les protocoles FTP et TFTP ne sont pas sécurisés, toutes les données transférées sont envoyées en texte clair. Ceci comprend les ID d'utilisateurs, les mots de passe ou le contenu des fichiers en texte clair. L'analyse de la session FTP de couche supérieure permet d'identifier rapidement l'ID d'utilisateur, le mot de passe ainsi que les mots de passe pour le fichier de configuration. L'analyse des données TFTP de couche supérieure est un peu plus complexe. Toutefois, le champ de données peut être examiné et les informations d'ID d'utilisateur et de mot de passe pour la configuration peuvent être extraites.

Tâche 7 : nettoyage

Au cours de ces travaux pratiques, plusieurs fichiers ont été transférés vers l'ordinateur hôte et doivent être supprimés.

Sauf indication contraire du formateur, mettez les ordinateurs hôtes hors tension. Enlevez le matériel utilisé durant les travaux pratiques, et préparez la salle pour le cours suivant.

