

AC11.04 Maîtriser les rôles et les principes fondamentaux des systèmes d'exploitation afin d'interagir avec ceux-ci pour la configuration et l'administration des réseaux et services fournis

Ressources/SAE : R108 R102 SAé1.01 SAÉ1.02

## **R1.02**

### **Contexte**

Dans le cadre de l'étude du fonctionnement du protocole FTP, une topologie réseau a été mise en place pour se connecter à un serveur et analyser les paquets échangés. Cette analyse permet de mieux comprendre les interactions entre les différentes couches du modèle OSI.

### **Savoir mis en œuvre**

- Connaissance des sept couches du modèle OSI (physique, liaison de données, réseau, transport, session, présentation et application).
- Fonctionnement du protocole FTP (File Transfer Protocol) pour le transfert de fichiers.

### **Savoir-faire mis en œuvre**

- Création d'une topologie réseau avec un client et un serveur FTP.
- Capture des paquets échangés à l'aide d'un outil d'analyse réseau (ex. Wireshark).
- Interprétation des données capturées en fonction des couches du modèle OSI.

### **Savoir-être mis en œuvre**

- Attention aux détails pour l'analyse des paquets capturés.
- Rigueur dans la configuration des équipements et des outils d'analyse.
- Patience et persévérance pour identifier et corrélérer les informations pertinentes.

### **Tâche réalisée et les résultats**

#### **1. Mise en place de la topologie réseau :**

- Un client FTP a été configuré pour se connecter au serveur situé à l'adresse IP `172.16.254.3`.
- Le serveur FTP est configuré pour répondre aux requêtes FTP et transférer des fichiers.

#### **2. Analyse des paquets capturés :**

- Première étape : Le client envoie une requête FTP pour récupérer le fichier `s1-central`.
- Paquet capturé : `Request: RETR s1-central`.
- Deuxième étape : Le serveur répond en ouvrant une connexion pour le transfert en mode binaire.
- Réponse capturée : `150 Opening BINARY mode data connection for s1-central (3100 bytes).`
- Troisième étape : Le serveur confirme la fin du transfert avec un message de succès.
- Réponse capturée : `226 File send OK.`

### 3. Corrélation avec les couches OSI :

- Couche application : Protocole FTP pour l'échange des commandes et des données.
- Couche transport : Utilisation du protocole TCP pour garantir la fiabilité de la connexion.
- Couche réseau : Adressage IP pour le routage des paquets entre le client et le serveur.
- Couche liaison de données et physique : Transmission des données sur le support réseau.

### Résultats obtenus

- Une connexion FTP fonctionnelle a été mise en place entre le client et le serveur.
- Les paquets capturés montrent clairement les étapes de la communication FTP, du début (requête) à la fin (confirmation de transfert).
- Une compréhension approfondie des interactions entre les couches OSI a été acquise grâce à l'analyse des paquets.

Voici l'analyse de la demande de connexion FTP :

#### 1. Les trois groupes d'unités de données associés au transfert sont :

242	7.709655	172.16.254.3	eagle-server.exempl...	FTP	71	Request: RETR s1-central
7c 57 58 d0 82 f4 00 24	c4 32 ee f8 08 00 45 00	WX....\$ .2....E.				
00 6e ea 59 40 00 3e 06	e8 74 c0 a8 fe fe ac 10	.n.Y@.>. .t.....				
fe 03 00 15 e7 8e 12 9e	97 a2 75 a6 d2 54 50 18	..... .u..TP.				
05 b4 28 23 00 00 31 35	30 20 4f 70 65 6e 69 6e	..(#..15 0 Openin				
67 20 42 49 4e 41 52 59	20 6d 6f 64 65 20 64 61	g BINARY mode da				
74 61 20 63 6f 6e 6e 65	63 74 69 6f 6e 20 66 6f	ta connec tion fo				
72 20 73 31 2d 63 65 6e	74 72 61 6c 20 28 33 31	r s1-cen tral (31				
30 30 20 62 79 74 65 73	29 2e 0d 0a	00 bytes )...				

  

246	7.712133	eagle-server.exempl...	172.16.254.3	FTP	124	Response: 150 Opening BINARY mode data connection for s1-central (3100 bytes).
7c 57 58 d0 82 f4 00 24	c4 32 ee f8 08 00 45 00	WX....\$ .2....E.				
00 6e ea 59 40 00 3e 06	e8 74 c0 a8 fe fe ac 10	.n.Y@.>. .t.....				
fe 03 00 15 e7 8e 12 9e	97 a2 75 a6 d2 54 50 18	..... .u..TP.				
05 b4 28 23 00 00 31 35	30 20 4f 70 65 6e 69 6e	..(#..15 0 Openin				
67 20 42 49 4e 41 52 59	20 6d 6f 64 65 20 64 61	g BINARY mode da				
74 61 20 63 6f 6e 6e 65	63 74 69 6f 6e 20 66 6f	ta connec tion fo				
72 20 73 31 2d 63 65 6e	74 72 61 6c 20 28 33 31	r s1-cen tral (31				
30 30 20 62 79 74 65 73	29 2e 0d 0a	00 bytes )...				

  

249	7.712485	eagle-server.exempl...	172.16.254.3	FTP	73	Response: 226 File send OK.
7c 57 58 d0 82 f4 00 24	c4 32 ee f8 08 00 45 00	WX....\$ .2....E.				
00 3b ea 5b 40 00 3e 06	e8 a5 c0 a8 fe fe ac 10	.;.[@.>. ....				
fe 03 00 15 e7 8e 12 9e	97 e8 75 a6 d2 54 50 18	..... .u..TP.				
05 b4 d9 a8 00 00 32 32	36 20 46 69 6c 65 20 73	.....22 6 File s				
65 6e 64 20 4f 4b 2e 0d	0a	end OK..				

## **Contexte**

Dans un environnement de machine virtuelle sous Linux Ubuntu, l'outil Python Scapy a été utilisé pour créer et analyser des paquets réseau. Cette expérimentation s'inscrit dans l'étude des couches du modèle OSI et de la structure des trames Ethernet.

### **Savoir mis en œuvre**

- Connaissance des couches du modèle OSI :
  - Application : Données.
  - Transport : Segment/Datagramme.
  - Internet : Paquet.
  - Accès réseau : Trame/Bit.
- Structure des trames Ethernet et fonctionnement des adresses MAC.

### **Savoir-faire mis en œuvre**

- Utilisation de Scapy pour la création et l'analyse de paquets.
- Identification et personnalisation des champs d'une trame Ethernet :
  - ``dst`` (destination) : l'adresse MAC de destination (par exemple, ``ff:ff:ff:ff:ff:ff`` pour une diffusion).
  - ``src`` (source) : l'adresse MAC source (par exemple, ``08:00:27:50:6f:1d``).
  - ``type`` : le type de protocole encapsulé dans la trame Ethernet (par exemple, ``0x0800`` pour IP).
- Interprétation des avertissements et erreurs dans Scapy.

### **Savoir-être mis en œuvre**

- Rigueur dans la saisie des commandes pour éviter les erreurs de syntaxe.
- Patience et persévérance pour interpréter les retours du terminal.
- Curiosité pour explorer les fonctionnalités avancées de Scapy.

### **Tâche réalisée et les résultats**

#### **1. Création d'une trame Ethernet :**

- La commande ``ma_trame = Ether()`` a été utilisée pour créer une trame Ethernet.
- La commande ``ma_trame.show()`` a affiché les détails de la trame, notamment :
  - ``dst`` : ``ff:ff:ff:ff:ff:ff`` (adresse de diffusion).
  - ``src`` : ``08:00:27:50:6f:1d`` (adresse MAC source).
  - ``type`` : ``0x9000`` (type de protocole encapsulé).

#### **2. Analyse des paquets :**

- Le message WARNING: Mac address to reach destination not found. Using broadcast. indique que l'adresse de destination n'a pas été spécifiée, ce qui a entraîné l'utilisation de l'adresse de diffusion (`ff:ff:ff:ff:ff:ff`).

### 3. Résultats obtenus :

- Une trame Ethernet a été générée avec succès en utilisant Scapy.
- Les champs `dst`, `src` et `type` ont été identifiés et peuvent être personnalisés selon les besoins.

- Application -> Données
- Transport -> Segment/Datagramme
- Internet -> Paquet
- Accès réseau -> Trame/Bit

```
>>> ma_trame = Ether()
>>> ma_trame.show()
WARNING: Mac address to reach destination not found. Using broadcast.
###[ Ethernet ]###
  dst      = ff:ff:ff:ff:ff:ff
  src      = 08:00:27:50:6f:1d
  type     = 0x9000
```

dst (destination) : l'adresse de destination du paquet IP ou MAC

src (source) : l'adresse source du paquet IP ou MAC

type : le type de protocole encapsulé dans la trame Ethernet (par exemple 0x800 pour IP)

## **R1.08**

### **Contexte**

Dans une infrastructure de machines virtuelles, une liaison interne a été configurée pour permettre la communication entre :

- Un serveur Ubuntu (`shemsserv`).
- Une machine Ubuntu client (`client-nfs`).
- Une machine Windows (`windows`).

Cette liaison utilise des adresses IP internes pour assurer la connectivité et permet d'établir une communication locale efficace.

### **Savoir mis en œuvre**

- Gestion des fichiers de configuration réseau (e.g., `/etc/hosts` sur les machines Linux).
- Compréhension des mécanismes de résolution de noms et de connectivité réseau interne.
- Configuration et tests de communication entre les machines.

### **Savoir-faire mis en œuvre**

- Modification et mise à jour du fichier `/etc/hosts` pour la résolution locale des noms.
- Configuration des adresses IP statiques pour les machines virtuelles.
- Tests de connectivité via des commandes réseau telles que `ping` et `nslookup`.

### **Savoir-être mis en œuvre**

- Rigueur dans la configuration des noms de domaine et des adresses IP.
- Organisation et documentation précise des paramètres réseau.
- Collaboration et partage des configurations entre les différentes machines.

### **Tâche réalisée et les résultats**

#### **1. Configuration du fichier `/etc/hosts` :**

- Sur les machines Ubuntu (serveur et client), le fichier `/etc/hosts` a été modifié pour inclure les entrées suivantes :

...

127.0.1.1 shemsserv shemsserv-nfs.domain.org

192.168.56.30 windows

192.168.56.20 client-nfs

...

- Cela permet une résolution de noms locale sans dépendre d'un serveur DNS externe.

## 2. Configuration des adresses IP :

- Le serveur Ubuntu (`shemsserv`) et le client Ubuntu (`client-nfs`) ont été configurés avec des adresses IP statiques :

- `shemsserv` : 127.0.1.1 (loopback).
- `client-nfs` : 192.168.56.20.
- `windows` : 192.168.56.30.

## 3. Tests de connectivité :

- Les tests de connectivité ont été effectués avec succès entre les machines :
  - La commande `ping windows` depuis le client Ubuntu a permis de vérifier la communication avec la machine Windows.
  - La commande `ping client-nfs` depuis le serveur Ubuntu a validé la communication interne.

## 4. Résultats :

- Les machines virtuelles communiquent correctement entre elles à l'aide des noms définis dans le fichier `/etc/hosts`.
- La résolution des noms est opérationnelle pour toutes les machines.

```
127.0.1.1 shemsserv shemsserv-nfs.domain.org
192.168.56.30 windows
192.168.56.20 client-nfs
```

## SAÉ1.02

### Contexte

Dans le cadre de l'administration réseau et de la gestion des services fournis, il est essentiel de comprendre et de maîtriser les principes fondamentaux des systèmes d'exploitation. Cela inclut :

- La gestion des rôles et des services offerts par le système d'exploitation.
- La configuration des paramètres réseau pour assurer une connectivité optimale.
- L'administration des services critiques tels que DHCP, et SSH.
- La gestion des sauvegardes via des outils appropriés pour garantir la disponibilité et la sécurité des données.

### Savoirs mis en œuvre

- Connaissance approfondie des rôles système tels que serveur DHCP et serveur SSH.
- Gestion des services pour l'administration et la configuration des réseaux.

## **Savoir-faire mis en œuvre**

### **1. Gestion des rôles système**

- DHCP : Configuration d'un serveur DHCP pour l'attribution automatique des adresses IP aux postes clients.
- SSH : Activation et sécurisation du protocole SSH pour l'administration à distance.

### **2. Configuration des paramètres réseau**

- Configuration des interfaces réseau avec des adresses IP statiques ou dynamiques selon les besoins.
- Vérification de la connectivité réseau et résolution des problèmes liés aux paramètres mal configurés.

### **3. Administration des services critiques**

- Déploiement d'un service DHCP :
  - Configuration des plages d'adresses (scopes) pour différents sous-réseaux.
  - Surveillance des baux DHCP et gestion des réservations d'adresse.
- Administration de SSH :
  - Gestion des clés publiques et privées pour un accès sécurisé.
  - Limitation des utilisateurs autorisés pour un contrôle renforcé.

## **Savoir-être mis en œuvre**

- Rigueur : Appliquer des méthodologies éprouvées pour garantir des configurations fiables.
- Autonomie : Capacité à effectuer les tâches d'administration en toute indépendance.
- Proactivité : Anticiper les besoins et les problèmes pour assurer la continuité des services.

## **Tâches réalisées et résultats**

### **1. Configuration d'un serveur DHCP :**

- Mise en place d'un serveur DHCP fonctionnel.
- Attribution automatique des adresses IP dans le sous-réseau 192.168.1.0/24.
- Gestion des réservations pour les équipements critiques.

### **3. Administration sécurisée via SSH :**

- Installation et configuration de SSH sur les systèmes Linux et Windows.
- Mise en place de clés SSH pour un accès sécurisé.
- Restriction des accès aux utilisateurs autorisés uniquement.

Tutoriel : Configuration de SSH et gestion des clés

Installation et activation de SSH

Gestion des clés SSH

## 1. Génération des clés :

- Sur le client : ``ssh-keygen -t rsa -b 2048``.

## 2. Copie de la clé publique sur le serveur :

- Utilisez la commande : ``ssh-copy-id user@192.168.1.1``.

