École de technologie supérieure
Département de génie logiciel et des TI

Trimestre : **Hiver 2024**
Enseignant : **Amine Boukhtouta**

Chargé de lab. : **Hafidh Zouahi**

GTI723 – Test d'intrusion

LABORATOIRE 1

Introduction

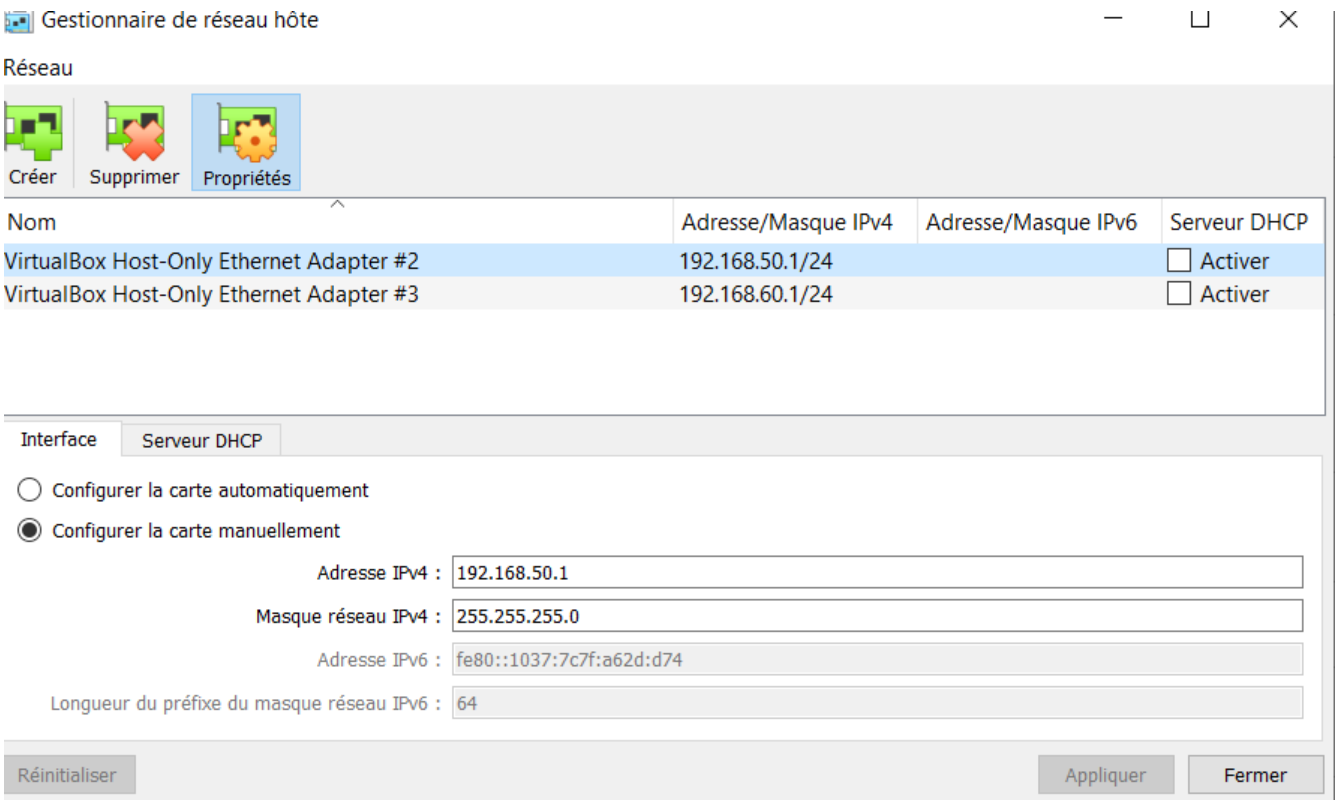
Internet a évolué dans les années 70, où la sécurité n'était pas parmi les spécifications initiales. Un pentester pourrait donc faire face à plusieurs vulnérabilités dans les protocoles de communication lors d'un test d'intrusion. De plus, plusieurs concepts réseau utilisés dans les différentes phases des tests d'intrusion nécessitent une compréhension des protocoles de communication. Il est donc important de bien saisir le fonctionnement de ces derniers, ainsi que les différents types de vulnérabilités.

Le but de ce premier laboratoire est de :

- Familiariser l'étudiant avec les faiblesses des protocoles de communication.
- Utiliser des concepts basés sur les protocoles de communication lors d'un test d'intrusion.

Configuration du réseau

1. Installez VirtualBox sur votre machine : https://www.virtualbox.org/wiki/Downloads
2. Téléchargez le fichier “Lab1 E.ova”.
3. Créez deux nouveaux segments réseau, en allant sur le menu « file ➔ Host network manager », et cliquez sur le bouton « Create ». Voir la figure suivante :



4. Attribuez la plage d’adresses 192.168.50.1/24 au premier segment crée.
5. Attribuez la plage d’adresses 192.168.60.1/24 au deuxième segment crée.
6. Assurez-vous que le serveur DHCP est désactivé sur les deux segments.
7. Importez les machines virtuelles “Lab1 E.ova” sur VirtualBox, en allant sur le menu « File ➔ Import Appliance ».
8. Dans la section « Network » de la configuration des machines : LabM1, LabM2, LabM3, LabM4, LabM5, et LabM6, allez dans l’onglet « Adapter1 », choisissez « Attached to Host-Only Adapter » et attachez l’interface au premier segment crée « VirualBox Host-Only Ethernet Adapter #2 ».
9. Dans la section « Network » de la configuration de la machine LabM3 allez dans l’onglet « Adapter2 », choisissez « Attached to Host-Only Adapter » et attachez l’interface au deuxième segment crée « VirualBox Host-Only Ethernet Adapter #3 ».

10. Dans la section « Network » de la configuration de la machine LabM8 allez dans l'onglet « Adapter1 », choisissez « Attached to Host-Only Adapter » et attachez l'interface au deuxième segment crée « VirualBox Host-Only Ethernet Adapter #3 ».
11. En allant dans : Settings ➔ Network ➔ Advanced; modifiez l'adresse MAC de chaque interface réseau sur l'ensemble des machines, comme suit :

    LabM1: 080027D07C89

    LabM2: 080027B15CEB

    LabM3: Interface 1: 080027A01599

    Interface 2: 08002733904C

    LabM4: 0800278A94BA

    LabM5: 0800273BEEE4

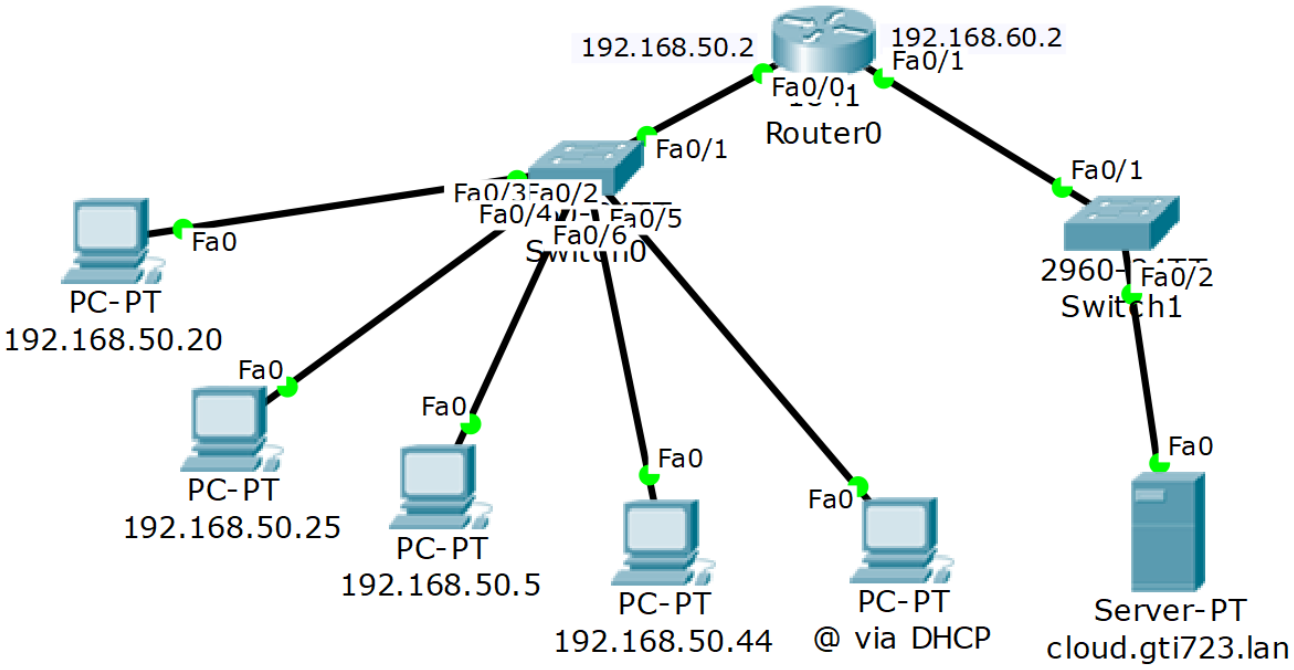
    LabM6: 080027A062D5

    LabM8 : 080027EAD5BA

12. Démarrez l'ensemble des machines virtuelles.
13. Connectez-vous sur la machine LabM2 (Kali Linux) :
    - Username : root
    - Mot de passe : GTI723!

Diagramme réseau

Le diagramme réseau du laboratoire est le suivant :



Remarques

- Pour chaque question vous devez inclure dans le rapport :
    - Le script (code développé)
    - Le résultat de l'exécution du script.

- Tous les scripts doivent être développés avec le langage de programmation Python.
- Tous les scripts doivent être développés sur la machine LabM2.

I. Partie 1 : Interaction avec les services via les protocoles de communication

Lors d'un test d'intrusion, le pentester pourrait identifier différents services et applications exposés sur le réseau. Il est donc important de savoir communiquer avec ces services afin de comprendre leurs fonctionnements. De plus le pentester pourrait se retrouver dans des environnements où les outils automatiques de test d'intrusion ne sont pas disponibles.

Dans cette première partie du laboratoire, vous devez utiliser la bibliothèque « socket » afin d'envoyer et de recevoir des données à travers l'interface réseau.

La machine **192.168.50.20** dispose d'un serveur d'application hébergé sur le port **723** utilisant le protocole **TCP**.

1. Implémenter un client TCP permettant de se connecter au serveur **192.168.50.20** sur le port **723**.
   a) Exécuter le script afin de se connecter au serveur, et récupérer les informations envoyées par ce dernier, sachant qu'il pourrait y avoir un délai d'une seconde (**1 s**) entre les segments TCP lors de l'émission de ces derniers par le serveur.
   b) Envoyer une commande permettant de récupérer le fichier de configuration.
   c) Envoyer une commande permettant de recevoir l'application distribuée par ce serveur d'application.
   d) Exécuter l'application reçue.

Certaines machines dans le réseau initient des communications **TCP** vers le port **723** de la machine **192.168.50.25**

2. Implémenter un serveur TCP permettant d'écouter sur le port **723**, et d'afficher les données transportées par les segments TCP envoyés par les clients.
   a) Exécuter le script, et récupérer les données reçues.
   b) Identifier la provenance de chaque donnée (adresse IP, et port source).
   c) Envoyer un segment TCP adapté à la donnée envoyée par chaque client.

La machine **192.168.50.20** héberge un service sous le port **7230** utilisant le protocole **UDP**.

3. Implémenter un client UDP permettant la communication avec le serveur **192.168.50.20** sur le port **7230**
   a) Exécuter le script afin de communiquer avec le serveur, envoyer la donnée « GTI723 » et récupérer les informations envoyées par ce dernier.

Certaines machines dans le réseau envoient des données à la machine **192.168.50.25** sur le port **7230** à travers des datagrammes UDP.

4. Implémenter un serveur UDP permettant d'écouter sur le port **7230**, et d'afficher les données transportées par les datagrammes UDP envoyés par les clients.
   a) Exécuter le script, et récupérer les données reçues.
   b) Identifier la provenance de chaque donnée (adresse IP, et port source).
   c) Envoyer un datagramme UDP adapté à la donnée envoyée par chaque client.

5. écrire un script permettant d'implémenter les fonctionnalités suivantes :

    a) Client UDP
    b) Client TCP
    c) Serveur TCP
    d) Serveur UDP
    e) Shell TCP
    f) Reverse Shell TCP
    g) Shell UDP
    h) Reverse Shell UDP
    i) Proxy TCP
    j) Proxy UDP

PS : Exemple d'utilisation du script pour créer un serveur TCP :
python script.py -t tcp -p 444
Exemple d'utilisation du script pour créer un Shell TCP :
python script.py -t tcp -p 444 -e
Exemple de connexion au Shell TCP:
python script.py -t tcp -i 192.168.50.25 -p 444

II. Partie 2 : Protocole ARP

1. Écrire un script permettant de sniffer les paquets sur une interface réseau, et d'afficher les paquets destinés à un port UDP donné.
2. Utiliser le script afin d'afficher tous les paquets destinés à la machine **192.168.50.25** sur le port UDP **7230**.

La machine **192.168.50.20** envoie des informations **syslog** via le protocole UDP sur le port **514** à la machine **192.168.50.44**.

3. Écrire un script permettant l'implémentation d'une réponse ARP.
4. En utilisant les deux scripts, intercepter et afficher les paquets UDP envoyés à partir de la machine **192.168.50.20** à la machine **192.168.50.44** sur le port **514**

La machine **192.168.50.20** initie des connexions TCP à la machine **192.168.50.44** sur le port **7723**.

5. Empoisonner le cache ARP des deux machines afin de pouvoir intercepter ces communications TCP.
6. En utilisant **Wireshark**, accéder aux paquets de cette communication.
7. Expliquer pourquoi la connexion TCP n'est pas complétée entre les deux machines.
8. Faites les modifications nécessaires afin de pouvoir intercepter toutes les communications entre les deux machines (sur le port **7723**).
9. Réimplémenter le script d'empoisonnement du cache ARP en utilisant la bibliothèque **scapy**.

III. Partie 3 : protocole IP

La machine **192.168.50.2** initie des connexions **HTTP** à la machine **192.168.50.20** sur le port **80**. Le port en question n'est accessible qu'à partir de l'adresse IP **192.168.50.2**

1. à partir de la machine **192.168.50.25** et en utilisant la bibliothèque **scapy**, initier une communication HTTP à la machine **192.168.50.20** sur le port **80**, tout en prétendant être la machine **192.168.50.2** (usurpation de l'adresse IP source).

IV. Partie 4 : protocole DNS

La machine **192.168.50.44** envoie une requête DNS au serveur **192.168.50.5** afin de résoudre le nom de domaine **cloud.gti723.lan.**

1. Sur la machine **192.168.50.25**, Intercepter cette requête DNS, et fabriquer une réponse DNS afin de forcer la machine **192.168.50.44** d'initier la connexion HTTP vers votre machine.
2. En utilisant le script développé dans la première partie, lancer un proxy sur votre machine écoutant sur le port **80** et permettant d'afficher le trafic HTTP entre la machine **192.168.50.44** et le serveur **192.168.60.77** (cloud.gti723.lan).

V. Partie 5 : protocole DHCP

Une machine **X** dans le réseau envoie des requêtes DHCP afin de solliciter le serveur **192.168.50.2**. Une fois que cette machine obtient une adresse IP, elle initie une communication FTP vers la machine **192.168.60.77**.

Le but dans cette partie du laboratoire est d'intercepter la communication FTP sachant que l'empoisonnement du cache ARP n'est pas possible.

En utilisant **scapy**:

1. Ecrire un script permettant de réserver toutes les adresses IP disponibles dans le serveur DHCP (**192.168.50.2**)
2. Ecrire un script permettant d'offrir une adresse IP a la machine **X**, ainsi qu'une adresse de passerelle afin de forcer le trafic FTP de passer par votre machine.

Pondération

Introduction ---------- 3%

Partie 1 : interaction avec les services via les protocoles de communication ----------**30%**

1. Client TCP
    a. Client TCP. **2%**
    b. Récupération du fichier de configuration. **2%**
    c. Récupération de l'application. **2%**
    d. Exécution de l'application reçue. **2%**
2. Serveur TCP
    a. Serveur TCP et récupération des données. **2%**
    b. Identification de la provenance. **2%**
    c. Envoie des segments TCP adaptés. **1%**
3. Client UDP
    a. Client UDP **2%**
4. Serveur UDP
    a. Serveur UDP et récupération des données **2%**
    b. Identification de la provenance. **2%**
    c. Envoie des datagrammes UDP adaptés. **1%**
5. Implémentation des fonctionnalités
    a. Client UDP **1%**
    b. Client TCP **1%**
    c. Serveur TCP **1%**
    d. Serveur UDP **1%**
    e. Shell TCP **1%**
    f. Reverse Shell TCP **1%**
    g. Shell UDP **1%**
    h. Reverse Shell UDP **1%**
    i. Proxy TCP **1%**
    j. Proxy UDP **1%**

Partie 2 : Protocole ARP----------**20%**

1. Implémentation du sniffer. **3%**
2. Interception des paquets. **1%**
3. Implémentation d'une réponse ARP. **4%**
4. Empoisonnement ARP et interception des paquets UDP. **2%**
5. Empoisonnement ARP et interception des paquets TCP. **2%**
6. accéder aux paquets de la communication. **2%**

7. Explication **2%**
8. Intercepter toutes les communications entre les deux machines **2%**
9. Réimplémenter le script d'empoisonnement en utilisant scapy. **2%**

Partie 3 : protocole IP----------**15%**

1. Usurpation de l'adresse IP. **15%**

Partie 4 : protocole DNS----------**15%**

1. Réponse DNS. **7.5%**
2. Interception du trafic. **7.5%**

Partie 5 : protocole DHCP----------**15%**

1. Réserver toutes les adresses IP disponibles. **7.5%**
2. Intercepter le trafic FTP. **7.5%**

Conclusions ---------- **2%**