Laboratoire 1

Mise en place du laboratoire

Le laboratoire « 1 » est constitué d'une maquette sous Docker contenant 4 conteneurs pré-configurées :

Machine	Nom de domaine	Configuration réseau	Applications et services
Serveur sous Debian 12	srvssh.local.sio.fr	Adresse IPv4 : 192.168.56.10/24 Passerelle : 192.168.56.254 Serveur DNS : 192.168.56.10	Service OpenSSH port 22/TCP Service DNS Bind port 53/UDP
Client sous Debian 12	clissh.local.sio.fr	Adresse IPv4 : 192.168.56.11/24 Passerelle : 192.168.56.254 Serveur DNS : 192.168.56.10	Environnement de bureau XFCE Service XRDP port 3389/TCP Client OpenSSH
Attaquant sous Kali Linux	kali.local.sio.fr	Adresse IPv4 : 192.168.56.12/24 Passerelle :192.168.56.254 Serveur DNS :192.168.56.10	Environnement de bureau XFCE Service XRDP port 3389/TCP Ettercap Git ssh-mitm Netfilter/Iptables
Routeur sous Debian 12	routeur.local.sio.fr	Adresses IPv4 : eth0 – DHCP eth1 -192.168.56.254/24 Serveur DNS : 192.168.56.10	Netfilter/Iptables

Toutes les machines sont accessibles en SSH (sur le port 22 à partir de l'hôte qui les héberge) et à partir de l'extérieur.

La machine Kali Linux et le client Debian sont également accessibles bénéficient d'une interface graphique accessible via un bureau à distance (protocole RDP sur le port 3389 à partir de l'hôte qui les héberge) et à partir de l'extérieur.

Voici les comptes, les mots de passe et les numéros de ports **accessibles de l'extérieur**, vous permettant d'accéder aux différentes machines virtuelles :

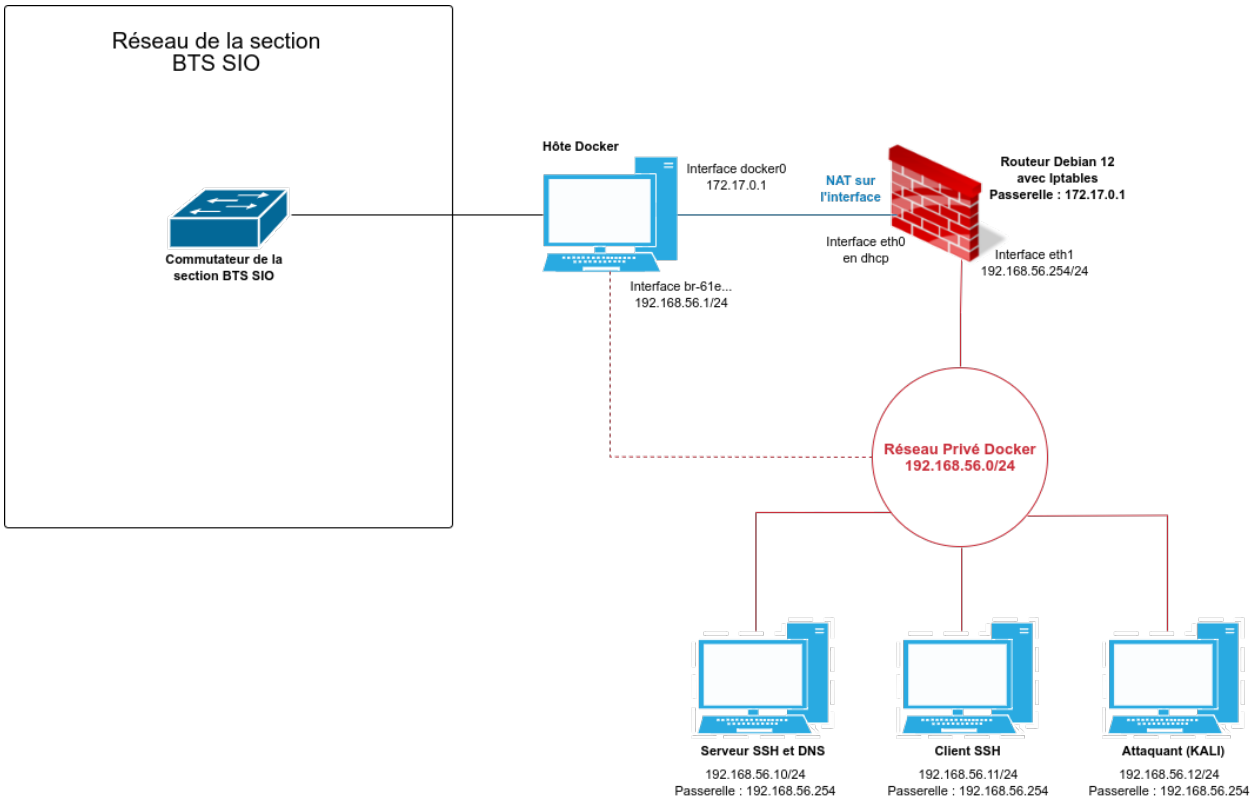
Intitulé de la machine	Nom d'utilisateur	Mot de passe	Ports SSH	Port RDP
Serveur SSH sous Debian 12	etusio	Fghijkl1234*	12222	
Client SSH sous Debian 12	etusio	Fghijkl1234*	22222	23389
Attaquant sous Kali Linux 2023.3	etusio	Fghijkl1234*	32222	33389
Routeur sous Debian 12	etusio	Fghijkl1234*	42222	

Lorsque des commandes nécessitant des privilèges administrateurs seront utilisées, il sera nécessaire d'utiliser la commande **sudo**.

```
etusio@srvssh:~$ sudo service ssh restart
```

Le professeur peut récupérer l'intégralité du laboratoire pour le personnaliser :
git clone https://forge.aeif.fr/btssio-labos-kali/lab1.git

Voici une représentation logique de la maquette proposée dans le cadre de ce laboratoire :



Explications sur les machines virtuelles

L'objectif de **Kali Linux** est de fournir une distribution basée sur Debian regroupant l'ensemble des outils nécessaires aux tests de sécurité d'un système d'information, notamment le test d'intrusion. L'intérêt de Kali Linux est de comporter près de 300 outils déjà installés pour travailler dans le domaine de la cybersécurité.

L'image docker est basée sur la version 2023.3 (kali-rolling). Seuls les paquets indispensables pour mener à bien l'activité ont été installées (dont les méta-paquets « core »). Mais il est possible d'effectuer des mises à jour ou d'installer de nouvelles applications. Elles seront persistantes tant que le conteneur n'est pas supprimé. Ce dernier peut bien sûr être stoppé.

Les machines sont basées sur Debian 12 (Bookworm) :

- **le routeur** fait office de routeur et de pare-feu avec iptables ;
- le client intègre une interface graphique accessible via le protocole RDP ;
- le serveur offre le service DNS.

Mise en œuvre de la maquette

La maquette ne peut être installée que sur une machine Linux (qui peut être une machine virtuelle sur VirtualBox par exemple).
Docker doit être installé sur l’hôte qui va héberger les conteneurs. **Le document 1 fournit une procédure**.

↗ Installer Docker et/ou vérifier qu’il soit bien lancé.
systemctl status docker

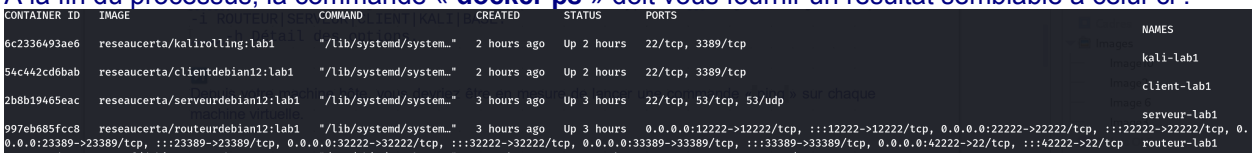
```
docker.service - Docker Application Container Engine
          Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
          Active: active (running) since Tue 2023-08-08 16:26:27 CEST; 48s ago
TriggeredBy: ● docker.socket
          Docs: https://docs.docker.com
          Main PID: 143 (dockerd)
          ...
```

↗ Récupérer le script « gestion_lab1.sh ».
wget https://forge.aeif.fr/btssio-labos-kali/lab1/-/raw/main/gestion_lab1.sh –output-document gestion_lab1.sh

↗ Lancer le script avec l’option « -c ».
bash gestion_lab1.sh -c

Le script récupère les images (si celles-ci ne sont pas déjà présentes sur l’hôte ou si elles sont plus anciennes que celles disponibles sur le docker hub) et lancent les conteneurs.

À la fin du processus, la commande « **docker ps** » doit vous fournir un résultat semblable à celui-ci :



```
CONTAINER ID   IMAGE                               COMMAND                  CREATED         STATUS        PORTS                                                                                                                NAMES
6c2336493ae6   reseaucerta/kalirolling:lab1        "/lib/systemd/system…"   2 hours ago     Up 2 hours    22/tcp, 3389/tcp                                                                                                     kali-lab1
54c442cd6bab   reseaucerta/clientdebian12:lab1     "/lib/systemd/system…"   2 hours ago     Up 2 hours    22/tcp, 3389/tcp                                                                                                     client-lab1
2b8b19465eac   reseaucerta/serveurdebian12:lab1    "/lib/systemd/system…"   3 hours ago     Up 3 hours    22/tcp, 53/tcp, 53/udp                                                                                               serveur-lab1
997eb685fcc8   reseaucerta/routeurdebian12:lab1    "/lib/systemd/system…"   3 hours ago     Up 3 hours    0.0.0.0:12222->12222/tcp, :::12222->12222/tcp, 0.0.0.0:22222->22222/tcp, :::22222->22222/tcp, 0.
0.0.0:23389->23389/tcp, :::23389->23389/tcp, 0.0.0.0:32222->32222/tcp, :::32222->32222/tcp, 0.0.0.0:33389->33389/tcp, :::33389->33389/tcp, 0.0.0.0:42222->22/tcp, :::42222->22/tcp   routeur-lab1
```

Depuis votre machine hôte, vous devriez être en mesure de lancer une commande « ping » sur chaque conteneur.

Commandes d’accès via SSH

Machine	Depuis la machine hôte	Depuis l’extérieur
Serveur	ssh etusio@192.168.56.10	ssh etusio@IP_hôte -p 12222
Client	ssh etusio@192.168.56.11	ssh etusio@ IP_hôte -p 22222
Kali	ssh etusio@192.168.56.12	ssh etusio@ IP_hôte -p 32222
Routeur	ssh etusio@192.168.56.254	ssh etusio@ IP_hôte -p 42222

Configuration du bureau à distance

À partir de l’hôte les conteneurs sont directement accessibles via leur adresse IP interne sur le port 3389.

À partir d’une machine externe les conteneurs sont accessibles via l’adresse IP de l’hôte sur les ports respectivement définis (23389 pour le client et 33389 pour l’attaquant Kali).

Pour le client REMMINA (sur Linux), une configuration supplémentaire doit être faite :
↗ Ouvrir les paramètres de connexion pour le profil de connexion.
↗ Accéder à « Avancé » et sélectionner « Cache des glyphes » et « Assouplir les vérifications des ordres ».

Persistance des données :

Au lancement de chaque conteneur un volume correspondant au dossier personnel de l'utilisateur « /home/etusio » est synchronisé avec un dossier (créé automatiquement) dans /var/lib/docker/volumes.

ls -l /var/lib/docker/volumes/

```
...
drwx-----x 3 user user     3 29 août 13:05 home_client_lab1
drwx-----x 3 user user     3 29 août 13:09 home_kali_lab1
drwx-----x 3 user user     3 29 août 13:00 home_routeur_lab1
drwx-----x 3 user user     3 29 août 13:00 home_serveur_lab1
```

Les données sont dans le sous-dossier « _data », par exemple :
ls -l /var/lib/docker/volumes/home_client_lab1/_data/

```
...
drwxr-xr-x 2 user user 2 30 août  12:17 Desktop
drwxr-xr-t 2 user user 2 30 août  12:17 thinclient_drives
…
```

Ces dossiers sont supprimés uniquement si le laboratoire est supprimé.

Le script permet également de :

- **stopper le laboratoire** : bash gestion_lab1.sh -s
- **lancer un laboratoire** préalablement stoppé : bash gestion_lab1.sh -l
- **redémarrer un laboratoire** : bash gestion_lab1.sh -r
- **supprimer un laboratoire** : bash gestion_lab1.sh -d

Lors de la suppression d'un laboratoire, tous les volumes et toutes les images associées sont supprimées. Vous pouvez rencontrer une erreur (sans conséquence) concernant la suppression de certaines images. Par exemple :
Error response from daemon: conflict: unable to remove repository reference "reseaucerta/serveurdebian12:lab2"

Ceci est normal si vous avez installé également le « lab2 » car le script tente de faire un nettoyage en profondeur et de supprimer toutes les images liées.

Il est nécessaire de stopper le laboratoire n°1 avant de lancer le laboratoire n° 2 et vice-versa.

Document 1 : Procédure d'installation de Docker

L'installation de Docker se réalise en quelques étapes très simples. Sur Linux, il suffit d'ajouter le dépôt Docker dans les sources de dépôts du système.

Installation des paquets nécessaires à l'utilisation du dépôt *docker* en https

```
apt update
apt install ca-certificates curl gnupg
```

Importation de la clé du dépôt docker

```
curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o /etc/apt/keyrings/docker.gpg
chmod a+r /etc/apt/keyrings/docker.gpg
```

Cette série de commandes télécharge (commande curl), déchiffre et décompresse (l'option --dearmor) le fichier GPG de clé, puis le stocke dans le répertoire /etc/apt/keyrings. Ensuite, elle ajoute des permissions de lecture à tous les utilisateurs sur ce fichier GPG.

Dans les commandes ci-après, des variables sont introduites, ce qui fait qu'elles sont valables pour n'importe quelle distribution sur Linux

Intégration du dépôt docker dans le fichier *source.list* et mise à jour des dépôts

```
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
tee /etc/apt/sources.list.d/docker.list > /dev/null

apt update
```

La variable « $(. /etc/os-release; echo "$ID") » renvoie la distribution (ici « debian »). Attention à l'espace entre « . » et le « / » et de même entre « add » et « - ».
La variable $(dpkg –print-architecture) renvoie l'architecture du processeur (ici amd64)

Installation de Docker

```
apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Docker est démarré (et est configuré pour démarrer automatiquement au lancement de la machine)

root@servDockerAR:~# systemctl status docker

```
docker.service - Docker Application Container Engine
         Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
         Active: active (running) since Tue 2023-08-08 16:26:27 CEST; 48s ago
TriggeredBy: ● docker.socket
         Docs: https://docs.docker.com
         Main PID: 143 (dockerd)
         ...
```

Enfin, pour ne pas avoir besoin d'utiliser docker en administrateur, il est possible de **modifier les droits de l'utilisateur en l'ajoutant au groupe docker.**

```
gpasswd -a votre_login docker
```

Il est nécessaire de se reconnecter avec cet utilisateur pour continuer à travailler avec Docker.