Projet 2A STI : Supervision et audit de la sécurité système dans un réseau

Jeremy Briffaut, ???

8 septembre 2014

1 Objectifs

Ce projet vous permettra de mettre en pratique vos connaissances acquises dans les différents modules de 1ère et 2ème année STI. Vous aborderez notamment l'administration et la sécurité système sous Linux ainsi que la programmation C système et réseau.
Vous travaillerez en binôme pour ce projet.

Ce projet couvrira les modules :

- administration système
- sécurité système
- réseau
- programmation C / SHELL
- commandes système
- cryptographie
- Base de Donnée
- XML

L'évaluation de fin de projet sera basée sur :

- une soutenance de 20 minutes (comprenant une démonstration de 10 minutes)
- un rapport détaillant ce que vous avez fait dans ce projet (une trame Latex vous sera fournie)
- une fiche que vous devrez remplir permettant d'évaluer les composants que vous avez abordés/implantés
- L'application que vous rendrez sous forme d'archive

Vous devrez rédiger le rapport au fur et à mesure du projet. Ce rapport devra notamment contenir une section par partie du projet décrivant sous forme de tutoriel ce que vous avez réalisé.

2 Documentation et Liens utiles

- `http://www.debian.org/`

— `http://ubuntu-fr.org/`

3 Description du projet

La supervision et l'audit de la sécurité dans un réseau sont un enjeu crucial pour les entreprises. L'objectif de ce projet est de concevoir une application client/serveur automatisant l'exécution de tâches d'audit/vérification (scans) de configuration de la sécurité système sous Linux. Vous implanterez par exemple les scans suivants :

— Information sur l'hôte (IP, hostname, distribution, version, ...)
— Intégrité des fichiers de certains répertoires
— Liste des paquets installés
— Liste des paquets à mettre à jours
— Liste des mises à jours critiques
— Dernières connexions (locales/distantes)
— ....

Chaque scan correspondra a un script écrit en langage `bash` exécuté localement sur l'esclave. Les résultats du scan seront ensuite remontés au maître.

Une application maître permettra de superviser un ensemble d'esclaves. Cette application permettra à un administrateur de lancer des scans sur les esclaves enregistrés ou d'afficher le résultats de ces scans. Cette application permettra d'avoir une vision globale de la sécurité du réseau.

Vous allez créer, pour ce projet, une application client/serveur en C fonctionnant dans un environnement GNU/Linux permettant de superviser et d'auditer la sécurité de machines clientes. Votre application sera composée de deux éléments :

— Un maître : élément central qui gère un ensemble d'esclaves, capable d'envoyer des requêtes d'exécution de scan aux esclaves, de récupérer les résultats des scans et de les afficher.
— Un esclave : reçoit des demandes de scans du maître, les exécute localement et envoie les résultats au maître

L'environnement de test que vous allez mettre en place sera constitué de :

— 1 machine virtuelle sous Debian (sans interface graphique) sur laquelle vous installerez votre maître
— 1 machine virtuelle sous Ubuntu (avec interface graphique) sur laquelle vous installerez votre esclave

3.1 Etapes du projet

Ce projet est divisé en différentes parties. Chaque partie fera l'objet d'une section a documenter dans votre rapport à rendre.

— Mise en place de l'environement de test : installation des 3 Machines Virtuelles (VMs) sous VMware

— Implémentation d'une application client/serveur en C : serveur/client multi-processus gérant plusieurs focntionnalités sur des ports différents (inscription, réception des résultats des scans, envoi de scans, inscription, ...)
— Gestion de l'inscription via le maître d'un esclave
— Gestion de l'inscription d'un esclave sur le maître
— Exécution d'un scan depuis le maître sur un esclave
— Réception et traitement des résultats d'un scan d'un esclave sur le maître
— Affichage des résultats d'un scan sur le maître
— Implémentation des scripts de vérification
— Stockage de la liste des esclaves dans une BDD
— Stockage des résultats dans une BDD
— Interface Web pour l'affichage des résultats
— Scans périodiques

4 Partie I : Installation des Machines Virtuelles

Pour mettre en place votre environnement de test, vous utiliserez VMWare Workstation qui est installé sur vos machines hôtes.

ATTENTION :

- dans VMWare, stockez vos VMs dans `/usr1` et non dans votre répertoire personnel pour ne pas dépasser votre quota
- pensez à sauvegarder régulièrement vos machines virtuelles sur un disque ou une clé USB
- installez vos machines virtuelles dans `/usr1/VOTRENOM/...` et fixer correctement les droits sur ce répertoire

4.1 Ce que vous devez faire

Vous installerez une machine qui vous servira de maître. Voici les caractéristiques de cette machine :

- dernière version stable de Debian
- sans interface graphique
- nom de la machine (`hostname`) : `serveur`

Vous installerez ensuite une machine esclave :

- dernière version stable de Ubuntu
- avec interface graphique (légère si possible)
- nom de la machine (`hostname`) : `client1`

Quelques remarques supplémentaires :

- Vous avez une totale liberté dans le choix des logiciels à installer (vim, gedit, ...), mais essayez de conserver un système minimal (ne pas installer KDE sur la machine serveur par exemple)
- Vous indiquerez dans votre rapport la liste des logiciels installés et comment vous les avez installés (`apt-get`, `aptitude`, ...)

4.2 Ce que vous devez rendre

Vous décrirez dans votre rapport comment vous avez installé ces 3 machines, en précisant notamment :

- Les différentes étapes de l'installations de vos machines
- Comment vous avez configuré vos machines

Vous pouvez illustrer votre documentation avec des captures d'écran

5 Partie II : Mise en place de l'environnement de développement

Afin de gérer votre projet et la phase de développement, vous installerez sur la machine serveur un logiciel de gestion de version. En vous aidant des différents tutoriaux disponible sur internet, vous devrez :

- installer un serveur GIT
- configurer le serveur GIT
- créer un projet pour héberger votre/vos programme(s)
- cloner le projet sur votre machine hôte

Je vous laisse le choix concernant l'environnement de développement (vim, eclipse, ...).

Vous pourrez ainsi développer depuis votre machine hôte et "pusher" votre code via GIT sur les machines serveur et client afin de le tester.

ATTENTION : pensez a cloner votre projet sur une clef USB de temps en temps ! ! !

6 Partie III : Client/Serveur en C

6.1 Architecture globale du client/serveur

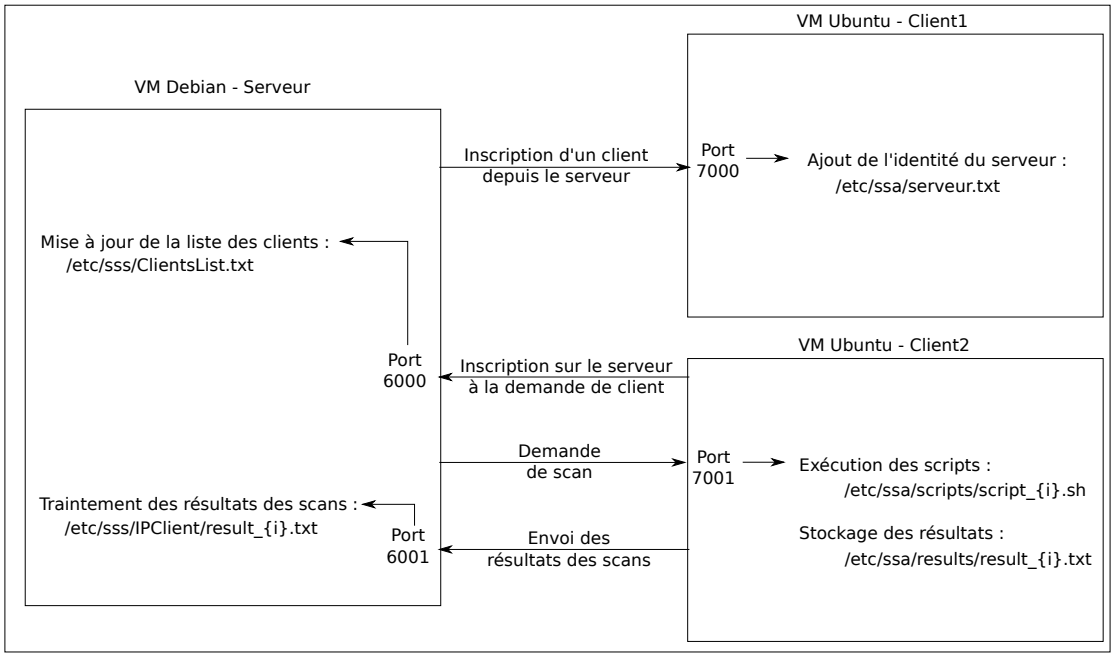

FIGURE 1 – Inscription via le maître d'un esclave

6.2 Inscription via le maître d'un esclave

L'objectif de cette partie est de permettre au maître d'inscrire un esclave. L'adresse IP de l'esclave à ajouter est fournie au maître et celui-ci contacte alors l'esclave pour récupérer les informations nécessaires.

Vous devrez :

— Fournir l'adresse IP de l'esclave au maître (la passer dans le fichier contenant la liste des esclaves)
— Contacter le client sur le port 7000 en envoyant les informations du maître
— L'esclave renvoie son hostname et la version de son agent
— Le maître récupère les informations et met à jour sa liste d'esclaves
— L'esclave met à jour le fichier avec les informations du maître

6.3 Inscription d'un esclave au maître

Cette partie permet à un esclave de s'inscrire auprès du maître s'il connait son adresse IP.

Ce que vous devez faire :

— Fournir l'adresse IP du maître à l'esclave

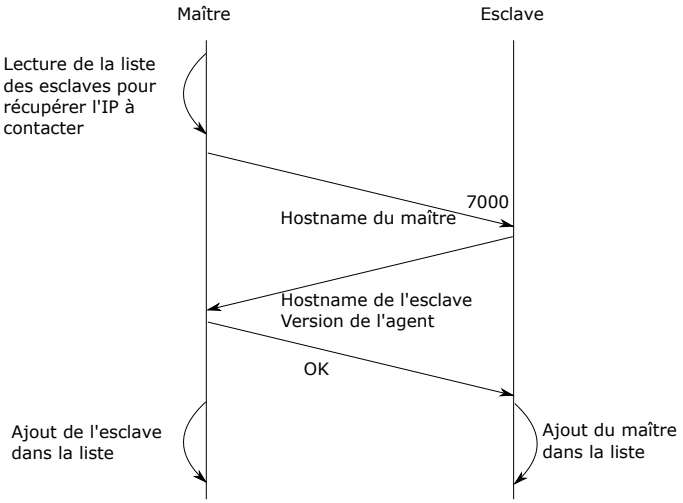

FIGURE 2 – Inscription via le maître d'un esclave

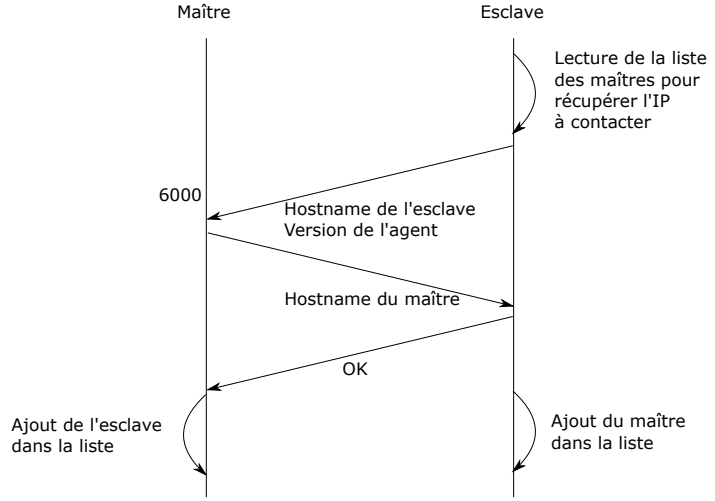

FIGURE 3 – Inscription d'un esclave au maître

— Envoyer une demande d'inscription au maître
— Inscrire l'esclave
— Envoyer la réponse du maître

Lors d'un inscription, n'oubliez pas de gérer le cas d'un esclave/maître déjà inscrit.

6.4 Lancement d'un scan depuis le maître

L'objectif de cette partie est de pouvoir lancer un scan sur un agent esclave depuis l'agent maître.

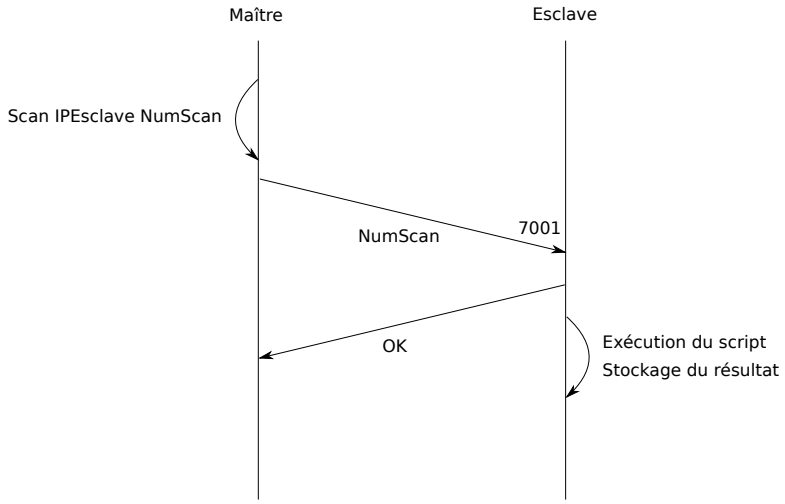

FIGURE 4 – Lancement d'un scan depuis le maître

Pour développer cette fonctionnalité, vous devrez :

- permettre d'envoyer une commande au maître lui donnant l'ip de l'esclave et un suméro de script (les scripts seront nommés `script_i.sh` où `i` est le numéro du script)
- envoyer à l'esclave concerné un message contenant le numéro du script à exécuter
- lancer l'exécution du script sur l'esclave
- stocker le résultat du script dans un fichier `resultat_i.txt` où `i` est le numéro du script

Vous pourrez implémenter des fonctionnalités supplémentaires, par exemple :

- pouvoir demander un scan au maître en donnant l'IP de l'esclave ou son hostname
- pouvoir demander l'exécution de l'ensemble des scripts présent sur un esclave
- pouvoir demander l'exécution d'un script sur l'ensemble des clients connus
- ...

Vous indiquerez dans votre rapport :

- comment vous demandez au maître de lancer un scan
- quel est le protocole que vous utilisez pour la communication entre le maître et les esclaves
- les fonctionnalités que vous avez implémentées

6.5 Réception et traitement des résultats d'un scan

Dans cette partie, vous devrez gérer la réception par le maître du résultat d'un scan.

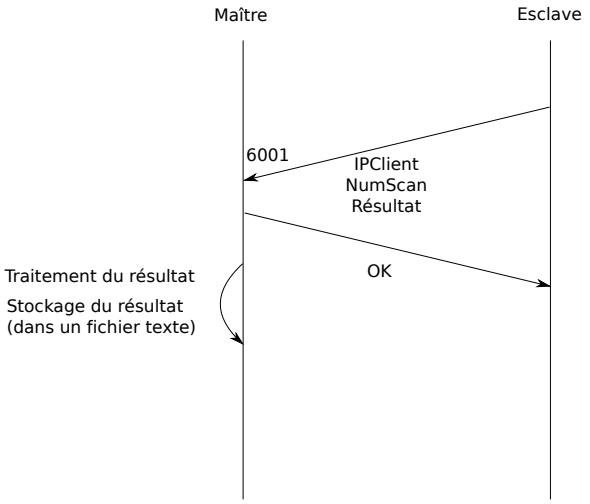

FIGURE 5 – Réception et traitement des résultats d'un scan

Pour gérer la réception du résultat d'un scan :

- l'esclave doit envoyer un message au maître contenant notament son IP, le numéro du script exécuté, et le résultat obtenu
- le maître doit réceptionner le message
- le maître doit stocker le résultat du scan. Les résultats seront stockés dans un répertoire associé au client (nommé avec son adresse IP).

Vous pourrez ajouter d'autres fonctionnalités :

- stocker les anciens résultats envoyés par le client (par exemple en utilisant le timestamp)
- ...

Vous indiquerez dans votre rapport :

- le protocole de communication entre l'esclave et le maître
- les éventuelles fonctionnalités que vous avez choisi d'implémenter

6.6 Affichage des résultats d'un scan

L'objectif de cette partie est de permettre la visualisation des résultats récupérés par le maître. La visualisation sera possible de deux manières :

- par un simple affichage texte dans la console
- en générant des pages HTML

Pour la première étape, vous devrez :

— implémenter la demande d'affichage des résultats des scripts d'un ou plusieurs clients
— afficher les résultats dans la console (par exemple sous forme de tableau)

6.7 Implémentation des scripts de vérification

Les scripts pouvant être exécutés sont stockés dans le répertoire `script` sur le client. Les scripts seront nommés `script_i.sh` où `i` est le numéro du script. Chaque fichier correspond à un script `bash` réalisant une analyse de la machine client et produisant une sortie qui devra être envoyée au serveur.

Voici un exemple de script pour lister les répertoires/fichiers sensibles (extrait du cours de sécurité système)

Listing 1 – script_0.sh

```
#!/bin/bash

# lister les fichiers avec le bit s
echo "list s"
find / \( -perm -4000 -o -perm -2000 \) -type f -exec ls -la {} \;

# lister les répertoires en écriture pour le groupe ou tout le monde
echo "list dir w"
find / -type d \( -perm -g+w -o -perm -o+w \) -not -perm -a+t -exec ls -lad {} \;
```

Voici une liste de script que vous pouvez implémenter :
— Remonter des informations sur l'hôte (hostname, ip, distribution, version, version du noyau, interface réseau, route, ...)
— Lister les comptes utilisateurs
— Calculer/vérifier l'intégrité de certains répertoires (`/bin`, `/sbin`, `/usr/bin`, `/usr/sbin`, `...`
— Lister les paquets installer, à mettre à jours, mise à jours critique, ...
— installer/configurer un paquet
— Dernières connections locale/distante
— Lister les services écoutant sur un port réseau
— Vérifier la configuration du pare-feu/déployer un pare-feu
— Tester le fichier `/etc/shadow` avec John The Ripper
— ...

7 Améliorations

Lorsque vous disposez d'un maître et d'un esclave fonctionnel, vous pouvez améliorer leur fonctionnement. Plusieurs amélioations sont proposées dans cette partie, mais vous pouvez aussi implémenter vos propres fonctionnalités.

7.1 SSL

Afin de sécuriser les échanges entre le maître et les esclaves, vous pouvez utiliser des sockets SSL et mettre en place une PKI pout gérer les certificats de vos VMs.

7.2 Sockets Unix

Pour faciliter la communication avec le maître ou l'esclave, vous pouvez utiliser des sockets Unix. De cette façon, vous pourrez par exemple demander facilement l'exécution d'un scan au maître.

7.3 Fichiers de service

Créez les scripts d'init pour le maître et l'esclave afin de faciliter la lancement et l'arrêt des programmes. Cela permettra également de lancer automatiquement le maître et l'esclave au démarrage de VMs.

7.4 Scans périodiques

Ajoutez la possibilité de demander une exécution périodique des scans. Par exemple, il sera possible de demander au maître d'exécuter le script `script_i.sh` sur l'ensemble des esclaves, ou bien d'exécuter tous les scripts disponibles sur l'esclave `IP_i`

7.5 Base de données

Vous pouvez utilisez une base de données afin de stocker les différentes informations présentes sur le maître. La BDD peut contenir le liste des esclaves connus par le maître et les résultats des différents scans réalisés.