Un environnement de laboratoire sûr pour la pratique d'enseignements de sécurité

Bruno Martin,
Université Côte d'Azur,
I3S-CNRS, BP 121,
06903 Sophia Antipolis, France.
Email : Bruno.Martin@univ-cotedazur.fr

***Résumé*—Cet article présente un retour d'expérience sur plusieurs années de travaux pratiques de sécurité en utilisant des ordinateurs et une salle de TP ≪ banalisée ≫. Par banalisée, nous entendons que les machines sont connectées sur un réseau public et que les étudiants ne disposent pas de droits administrateur sur la machine physique. Pour ce faire, nous virtualisons un hyperviseur de type 1 qui héberge plusieurs machines virtuelles dont nous décrivons le fonctionnement, les avantages et les inconvénients de cette solution à bas coût.**

I. Introduction

Utiliser les outils proposés dans une distribution d'audit de sécurité ou de tests d'intrusion impose un certain nombre de contraintes. Son utilisateur doit avoir les droits d'administrateur sur la machine pour faire du *packet sniffing* ou des attaques de l'homme du milieu. Plusieurs solutions sont envisageables. La première solution utilise des machines physiques et immobilise une salle dédiée, qui est séparée du réseau local et qui doit avoir en outre un minimum d'équipement réseau (a minima un ou plusieurs commutateurs, voire un serveur pour filtrer les accès au réseau Internet). Une seconde solution utilise des hyperviseurs de type 2 (par des logiciels de virtualisation comme VirtualBox ou VMware workstation) pour héberger quelques machines virtuelles. La troisième solution utilise également un hyperviseur de type 2 qui, au lieu d'héberger des machines virtuelles, exécute un hyperviseur de type 1 (comme VMware ESXi ou XenServer). Dans cet article nous dicutons les avantages et les inconvénients de ces trois solutions qui ont été testées ou utilisées pour des travaux pratiques de réseau ou de sécurité. Nous donnons ensuite un aperçu des travaux pratiques qui ont été réalisés avant de conclure en offrant trois points de vue : celui de l'étudiant (ou de l'utilisateur), celui de l'enseignant et celui de l'administrateur de la salle. Nous terminons cet article en présentant d'éventuelles améliorations avec l'utilisation d'un serveur de virtualisation.

II. Utiliser une salle dédiée

J'ai commencé à faire de travaux pratiques en ayant à disposition une salle dédiée. Cette possibilité a été utilisée pendant 10 ans (entre 2000 et 2009). La salle disposait de 24 ordinateurs dont la moité avait deux interfaces réseaux (pour réaliser une manipulation d'IP forwarding et d'IP masquerading), de deux commutateurs et d'un petit serveur configuré comme une passerelle entre le réseau du site et le réseau local de la salle (sur une adresse réseau non routable). Cette passerelle hébergeait également tout un ensemble de services : DNS, CVS, tftp (pour reconfigurer les commutateurs), NTP, proxy, serveur de réinstallation par kickstart. Son firewall n'autorisait que les connexions `http` et `https` à partir du réseau local. L'équipe système avait abandonné la maintenance de cette salle qui est devenue obsolète et n'a pas été renouvelée.

Parallèlement, je m'étais intéressé à la virtualisation pour la préparation des travaux pratiques sans avoir besoin de me déplacer dans la salle et j'ai donc étudié une solution de virtualisation (avec un hyperviseur de type 2).

III. Avec un hyperviseur de type 2

Un hyperviseur de type 2 est un logiciel qui est exécuté sur la machine physique hôte. Ce logiciel (VirtualPC à l'époque puis VirtualBox ou les outils VMware –workstation sous linux ou windows ou fusion sur MacOS–) émulent une architecture matérielle et permettent de lancer plusieurs systèmes d'exploitation invités sur l'hôte. Ils autorisent une mise en réseau selon trois modes de fonctionnement principaux :

NAT : ou traduction d'adresse réseau où la machine virtuelle accède au réseau comme si elle était connectée à un routeur hébergé par l'hôte.

Bridge : ou pontage qui partage l'interface physique de l'hôte. La machine virtuelle acquiert sont IP par un serveur externe à l'hôte.

Host Only : dans lequel la machine virtuelle ne peut communiquer qu'avec d'autres machines virtuelles comme si elles étaient reliées par un commutateur.

Ce type d'environnement permet de réaliser toutes les manipulations. Il suffit de virtualiser un petit serveur sur lequel on installe des services à auditer (typiquement un serveur web, un serveur `smtp`, `pop` ou `imap`). Un seconde machine virtuelle est installée avec une distribution d'audit de sécurité (comme Kali linux, Parrot OS ou BlackArch). Les deux machines virtuelles sont interconnectées en NAT pour accéder à Internet et partagent des adresses sur un réseau non routable virtualisé par l'hôte qui sert de routeur.

L'inconvénient principal de ce type d'environnement est que l'utilisateur doit avoir les droits d'administrateur sur la machine hôte pour autoriser le logiciel de virtualisation à

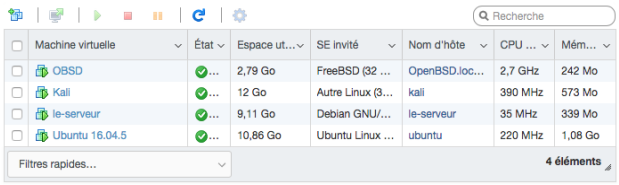

FIGURE 1. Interface de gestion des VM.

permettre le passage des cartes réseau virtuelles en mode de promiscuité. Rapellons que le mode de promiscuité permet à la carte réseau qui émule un réseau de type ethernet d'accepter tous les paquets qu'elle reçoit, même s'ils ne lui sont pas destinés. Ce mode est utilisé par les outils de *packet sniffing* comme `wireshark` ou `ettercap`. Selon le logiciel de virtualisation utilisé, il faut donc que l'utilisateur ait non seulement les droits d'administrateur sur les machines virtuelles mais aussi sur la machine hôte. Ce type de virtualisation est donc acceptable pour un ordinateur personnel (sous l'hypothèse de réaliser des travaux pratiques sur la base du BYOD) ou pour faire la mise au point des énoncés.

J'ai donc cherché un moyen d'≪ empiler ≫ les couches de virtualisation pour que l'utilisateur dispose des droits d'administrateur sur un ensemble de machines virtuelles ainsi que sur un réseau mais n'ait pas ces droits sur la machine hôte.

IV. Avec un hyperviseur de type 1 virtualisé

Un hyperviseur de type 1 est un noyau système léger et optimisé pour la gestion de noyaux de systèmes d'exploitation invités. Il y a beaucoup moins de fournisseurs qui offrent ce type de fonctionnalité. Principalement les produits de VMware –ESXi– et de Citrix –XenServer–. Pour des raisons historiques, je me suis concentré sur les produits VMware (le seul offrant les fonctionnalités au début des années 2000). Ce type d'hyperviseur n'embarque pas de système d'exploitation. Il comprend seulement un noyau linux qui est utilisé pour charger tout un ensemble de services de virtualisation. Le plus important d'entre eux est un noyau virtuel (le `vmkernel`). Celui-ci est au cœur de l'infrastructure. Après le chargement d'interfaces qui virtualisent le matériel, il permet d'héberger plusieurs noyaux de systèmes d'exploitation invités dans un mode client/serveur. Notons que VMware propose une version gratuite de son hyperviseur avec cependant quelques limitations : il n'est possible d'utiliser que 2 CPU de la machine physique et on ne peut virtualiser que 8 machines virtuelles. Il n'est pas possible d'administrer le serveur ESXi par les outils professionnels de vCenter qui contient les outils de gestion d'un entrepot de données et d'un (gros) ensemble d'hyperviseurs.

Normalement, l'hyperviseur de type 1 (ESXi) est utilisé directement sur le matériel d'une machine physique (typiquement un serveur). Dans le cas de notre environnement, nous avons décidé de virtualiser le serveur ESXi au moyen d'un hyperviseur de type 2 (VMware Workstation sous windows ou linux, Fusion sur MacOS). Pour en assurer la performance,

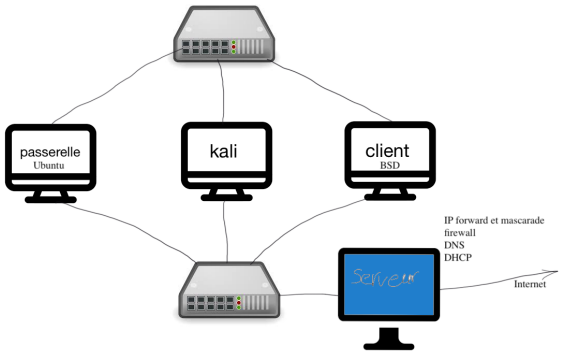

FIGURE 2. Réseaux émulés.

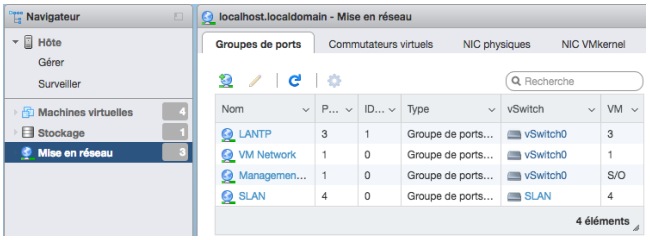

FIGURE 3. Mise en réseau des VM.

nous allouons 2 cœurs et 4,5 Go de RAM au serveur ESXi avec une mise en réseau de type NAT. Autrement dit, le serveur ESXi partage l'adresse IP de la machine hôte sur le réseau externe.

Le serveur ESXi virtualisé est lancé comme n'importe quelle machine virtuelle. A l'issue de son démarrage, il offre une adresse de connexion sur un réseau non routable (p.e. `192.168.210.139`. La machine physique fait office de routeur et prend la première adresse machine de ce réseau. Depuis la version 6 d'ESXi, il est possible de se connecter au serveur ESXi au moyen d'un navigateur web. La connexion se fait après une page d'authentification (plusieurs utilisateurs avec des rôles différents sont possibles). On arrive ensuite sur une page d'accueil qui permet de gérer le serveur ESXi et les machines hébergées (voir FIGURE 2).

A. *Configuration et installation*

L'intérêt d'utiliser un serveur ESXi est de pouvoir définir différents rôles d'utilisateurs (a minima un administrateur et un utilisateur) et de pouvoir réaliser une mise en réseau. La figure 3 décrit la mise en réseau de l'environnement. On y distingue une interface de gestion (celle sur laquelle on peut se connecter par le navigateur web) ainsi que plusieurs autres réseaux, interconnectés par des commutateurs virtuels (vSwitch0 et SLAN). Un réseau un peu particulier est VM Network qui correspond au réseau partagé par les machines virtuelles hébergées. Le gestionnaire du serveur ESXi nous permet d'autoriser le mode de promiscuité sur les nouveaux réseaux, l'usurpation d'adresses MAC et la transmission de paquets modifiés.

A présent, il suffit de « virtualiser » la configuration qu'on avait avec la salle dédiée, notamment sur les configurations réseau pour sécuriser l'environnement. On installe donc une machine virtuelle `le-serveur` qui interconnecte le réseau « externe » de gestion sur une de ses interfaces et le réseau partagé par l'interface principale des autres machines qu'on va virtualiser. Cette machine servira de passerelle entre l'interface montée en NAT de l'hyperviseur de type 2 et le commutateur virtuel LAN qui relie les machines virtuelles sur un réseau appelé SLAN. `le-serveur` fait donc de l'IP masquerading et de l'IP forwarding. Il filtre également les paquets qui sortent de SLAN pour n'autoriser que les requêtes `http` et `https`. Ces protocoles suffisent pour aller chercher des informations sur Internet et réaliser les opérations de mise à jour des paquets. La machine `le-serveur` héberge également un serveur de noms (`bind9`) et un serveur `dhcp` pour offrir aux machines virtuelles un accès internet sur un réseau commuté.

Afin d'assurer une meilleure sécurité de l'environnement, on fait également appel à la notion de rôles proposée par le serveur ESXi. Outre le rôle par défaut d'administrateur, on ajoute un nouveau rôle utilisateur auquel on donne des droits limités : autorisation d'allumer et d'éteindre le serveur ESXi, interdiction d'accéder à la machine `le-serveur`, interdiction de modifier les paramètres des réseaux virtualisés.

On peut alors installer et configurer les machines proposées aux utilisateurs : on ajoute alors trois nouvelles machines virtuelles (avec les gestionnaires `vmware-tools`). L'interface primaire de ces machines est reliée au vSwitch SLAN, chaque machine obtient une adresse IP sur le réseau SLAN par `dhcp` auprès de `le-serveur` ainsi qu'une entrée `dns` sur le réseau dédié. Leur interface secondaire est connectée sur le réseau `LANTP` par l'intermédiaire du vSwitch0 sur un réseau de type "Host Only". Le but principal des TP étant de réaliser des attaques entre un client et un serveur, il a été nécessaire d'avoir 3 machines : une pour servir de "serveur" (dans la dernière version, il s'agit d'une distribution `Ubuntu 16.04` par souci de simplicité), du client (pour montrer qu'il n'y a pas que linux, j'ai choisi d'utiliser un OpenBSD 6.2) et d'une machine d'attaque (je suis passé de la Backtrack 5 à une Kali linux 2 au fil du temps).

L'ensemble des machines a été configuré pour démarrer automatiquement ; d'abord `le-serveur` (qui doit être lancé à l'avance pour offrir des adresses et du routage aux autres machines) puis les trois autres machines.

B. Utilisation de l'environnement

Une fois tout installé et bien configuré, l'utilisateur se connecte sur la machine physique, démarre le serveur ESXi comme une machine virtuelle classique par VMware Workstation sous windows ou linux ou par VMware fusion sous MacOS [1]. Une fois le serveur ESXi lancé, il suffit de s'y connecter comme utilisateur au moyen d'un navigateur web pour accéder aux mavhines virtuelles. Dans une première version de cet environnement, sous ESXi 4.5, il fallait utiliser un client lourd `vSphere` qui n'était disponible que sous Windows. L'inconvénient d'utiliser notre environnement par l'interface web est que celui-ci a parfois un peu de mal à gérer les événements clavier (français) et souris. Il est préférable d'utiliser l'utilitaire VMware Remote Console qui fonctionne mieux pour la gestion des périphériques des machines virtuelles hébergées au sein du serveur ESXi.

L'avantage principal de cet environnement est de permettre aux utilisateurs de pouvoir travailler sur les 3 machines virtuelles hébergées par le serveur ESXi sans avoir les droits administrateur ni de la machine hôte, ni du serveur ESXi. En revanche, les utilisateurs disposent des droits administrateur sur les 3 machines virtuelles et peuvent également observer les 2 réseaux émulés ainsi qu'y faire de l'injection de paquets.

Un autre avantage de l'environnement est qu'il est (pour l'instant) entièrement gratuit et qu'il offre de réelles conditions de sécurité pour mettre en pratique des attaques impossibles à réaliser sur des machines physiques sur un réseau public. Les hôtes physiques peuvent de plus être utilisés comme des machines banalisées pour les autres TP sans avoir recours à un administrateur système, que ce soit pour déconnecter la salle ou réaliser des opérations d'administration sur ceux-ci.

L'inconvénient majeur réside dans le déploiement de l'environnement. Dans sa dernière version, même en allégeant au maximum les configurations logicielles des machines virtuelles, il requiert une trentaine de Giga-octets, ce qui est assez long à recopier, que ce soit en utilisant une clé USB ou par le réseau.

Un nouvel inconvénient est que les machines virtuelles hébergées dans le serveur ESXi –lorsqu'il est virtualisé– doivent obligatoirement être sur une architecture 32 bits qui a tendance à disparaitre des distributions. La raison principale évoquée est qu'il n'y a pratiquement plus d'architecture physique 32 bits et que les maintenir devient de plus en plus compliqué. (`Debian` maintient le support 32 bits).

V. Avec un serveur de virtualisation

L'idéal serait d'avoir accès à un serveur de virtualisation physique et de virtualiser toutes les machines virtuelles sur une même plate-forme. Il est préconisé de ne pas dépasser trois machines virtuelles par cœur. Aussi, pour réaliser des TP pour une vingtaine d'utilisateurs qui disposeraient chacun des mêmes trois machines virtuelles, il faudrait un serveur avec 64 cœurs (ce qui est proposé par différents fournisseurs) et un minimum de 64Go de RAM. Avec une telle configuration, il serait également possible d'ajouter facilement une machine test qui présenterait des vulnérabilités et qui serait accessible par tous les utilisateurs.

L'avantage serait d'être dispensé du déploiement des machines qui nécessite environ 1 homme.jour de travail dans une salle banalisée (pour une fréquence annuelle).

L'inconvénient, outre le coût, est de disposer d'une fraction de temps ingénieur pour l'administration du serveur, même si la gestion des machines virtuelles pourrait être déléguée aux enseignants responsables.

1. Le démarrage du serveur ESXi sous Fusion demande le mot de passe administrateur de la machine pour activer le réseau.

Une telle configuration permettrait aux utilisateurs de pouvoir accéder à leurs machines à partir d'un client logiciel depuis n'importe où par l'intermédiaire d'un VPN dédié.

VI. Exemples d'expériences réalisées

La configuration actuelle (qui fonctionne sur des machines sous Windows 10, disposant de 6Go de RAM, d'un processeur Intel i5 à deux cœurs et d'un disque mécanique) permet de réaliser tout un ensemble de manipulations dont nous donnons ci-dessous un aperçu. L'enseignement dispensé est composé de 2 cours magistraux pour expliquer les TP et d'une série de 5 TP de 4h. Le premier TP porte sur la réalisation d'une passerelle pour faire de l'`IP-forwarding` et de l'`IP-masquerading` entre son interface principale et son interface secondaire. On peut ainsi réaliser un réseau avec les interfaces secondaires des machines de l'utilisateur. On en profite pour installer un serveur `telnet`, mettre en place un serveur `ssh` qui autorise les connexions distantes et activer les services de `logs` offerts par `(r)syslog`. Le serveur `telnet` sera utilisé ultérieurement pour renifler la transmission d'un login non sécurisé. L'utilitaire `nmap` est utilisé pour la découverte du réseau et des services qui sont offerts par les machines. Le second TP commence par renifler le couple login/password transmis lors d'une connexion `telnet` entre un client et le serveur. On installe ensuite un serveur `apache` avec `openssl` en compilant les sources. On utilise ce serveur pour afficher une page de connexion comparable à une authentification style webmail, d'abord en `http` puis en `https`. Les couples login/password sont interceptés avec `ettercap` ou `bettercap` en réalisant une attaque de l'homme du milieu sur la transmission du certificat auto-signé. On met ensuite en place les firewalls sur les machines pour assurer une meilleure sécurité (via `netfilter` sous `linux` et `packet filter` sous `OpenBSD`). Un troisième TP s'intéresse au courrier électronique. On installe et on configure `postfix` sous `linux` (en profitant du champ `MX` du serveur de noms du serveur) et `OpenSMTP` sous `OpenBSD`. On essaye (selon que la configuration est en `smtp` ou `smtps`) d'écouter le trafic `smtp`. Un serveur `imap(s)` est installé puis attaqué. Les firewalls des machines prennent en compte ces nouveaux services. Dans le TP4, on configure un serveur `openvpn` en mode routé (qui travaille au niveau de la couche réseau pour relier deux machines) puis en mode ponté (qui travaille sur la couche de liaison et relie des réseaux distants). Les firewalls des machines prennent en compte ces nouveaux services. Le dernier TP permet de réaliser un mini audit de sécurité des machines qui ont été utilisées en utilisant le scanner de vulnérabilité `OpenVAS`. J'ai utilisé avec succès `Nessus` jusqu'au moment où cet outil est sorti du monde libre. Cependant, ces deux scanners de vulnérabilités partagent presque la même base de connaissance. On découvre aussi le framework `Metasploit` en essayant d'intégrer `OpenVAS` dans `Metasploit`. Il est regrettable que par manque de place, je n'ai pas pu installer de machine vulnérable à `Metasploit`.

Dans cet ensemble de TP, les étudiants apprennent à utiliser un hyperviseur de type 1 et à gérer des connexions distantes à un petit parc de machines sur un réseau virtuel isolé. Ils découvrent aussi comment installer des services (que ce soit par un gestionnaire de paquets ou par la compilation des sources), le fonctionnement des mécanismes de `logs`, comment créer des utilisateurs et séparer leurs privilèges. Ils prennent également en main une distribution d'audit de sécurité et de quelques uns des nombreux outils qui y sont intégrés. La configuration des réseaux virtuels commutés les force à réaliser de l'`arp poisonning` pour utiliser les outils de reniflage. Ils comprennent aussi le risque lié à l'utilisation de certificats auto-signés et la gestion d'une petite infrastructure de PKI.

A l'issue des TP, il leur est demandé de rédiger un compte-rendu sur l'ensemble des TP en proposant une politique de sécurité cohérente sur les services installés (la rédaction des TP et leurs choix durant les séances peuvent donner lieu a de multiples variantes).

Enfin, ils découvrent les deux environnements `linux` et `BSD` avec leurs avantages, leurs inconvénients et leurs différences.

VII. Point de vue des utilisateurs et évolution possible

Les paragraphes précédents ont présenté un environnement de laboratoire sûr pour la pratique des enseignements de sécurité pour l'utilisateur. Qu'en est-il pour l'administrateur de site et pour l'enseignant ? L'enseignant réalise et met à jour l'environnement chaque année. Il dispose des droit d'administrateur sur le serveur `ESXi` et sur la machine virtuelle qui sert de serveur (inaccessibles aux étudiants hormis pour allumer et éteindre). L'enseignant ne dipose pas des droits d'administrateur sur les machines physiques. C'est l'administrateur de site qui recopie le fichier du serveur `ESXi` sur les hôtes physiques, qui installe l'hyperviseur de type 2 (cela peut être VMware Workstation ou VirtualBox) et le client qui autorise la connexion sur les machines virtuelles hébergées par le serveur `ESXi`. Même en automatisant ces différentes tâches, le temps nécessaire au déploiement de l'environnement est de l'ordre de la journée. L'avantage est que la salle est banalisée et que ni l'enseignant, ni les étudiants n'ont besoin des mots de passe d'administration.

En outre l'environnement permettrait de réaliser d'autres TP (mettre en place un serveur de noms `dnssec` secondaire, faire des attaque du serveur `dns`, des attaques XSS, etc).

Il serait toutefois agréable de pouvoir disposer d'un serveur de virtualisation qui permettrait de créer plusieurs véritables réseaux locaux avec toutes les machines et en y incluant des machines vulnérables. Cela permettrait de réaliser des défis en plusieurs équipes qui pourraient attaquer et défendre leur parc de machines, voire disposer de plusieurs réseaux avec des configurations différentes. Un autre avantage serait de permettre aux étudiants de gérer des *snapshots* de leurs machines virtuelles pour tester une configuration et revenir éventuellement à la configuration précédente.