h e p i a
Haute école du paysage, d'ingénierie
et d'architecture de Genève

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences
Western Switzerland

Virtualisation & Sécurité

Cette étude, financée par le RCSO-TIC[1], s'est déroulée de septembre 2010 à janvier 2012
Elle s'inscrit dans le cadre du projet Visag
http://gridgroup.hefr.ch/visag

Bref descriptif du cahier des charges pour hepia[2]
http://www.tdeig.ch/vmware/Visag.pdf

Objectifs

Ce document tente de répondre aux questions suivantes :

- Comment aborder la sécurité d'une architecture virtualisée ?
- Peut-on réutiliser l'expérience du monde physique ?
- La virtualisation peut-elle augmenter la sécurité d'un service ?
- Quels sont les principaux risques liés à la virtualisation ?
- Quelles sont les bonnes pratiques ?
- Quels sont les avantages et les inconvénients du produit gratuit VMware ESXi 4.0 ?

Structure du document articulée autour des 5 parties suivantes :

Partie 1 : Contexte général et donnée du problème

Partie 2 : Analyse, rappel de sécurité et virtualisation

Partie 3 : Best Practices génériques & référentiel

Partie 4 : Analyse de sécurité de cette plateforme basée sur l'hyperviseur ESXi 4.0 rédigée par Cédric Penas

Partie 5 : Etat de l'art
Synthèse de diverses publications permettant d'identifier les principaux risques
D'un point de vue chronologique, cette partie résume les études préliminaires effectuées par l'auteur dans ce projet.
Le lecteur curieux commencera donc par cette partie qui mentionne divers liens très utiles.
Il trouvera également l'avis personnel de l'auteur de cette publication

Divers

Il n'est pas prévu de traduire ce document si bien que certains mots y figurent en anglais afin de faciliter son indexation sur internet.

J'en profite pour présenter et remercier les personnes du labo[3] qui m'ont aidé dans cette étude :

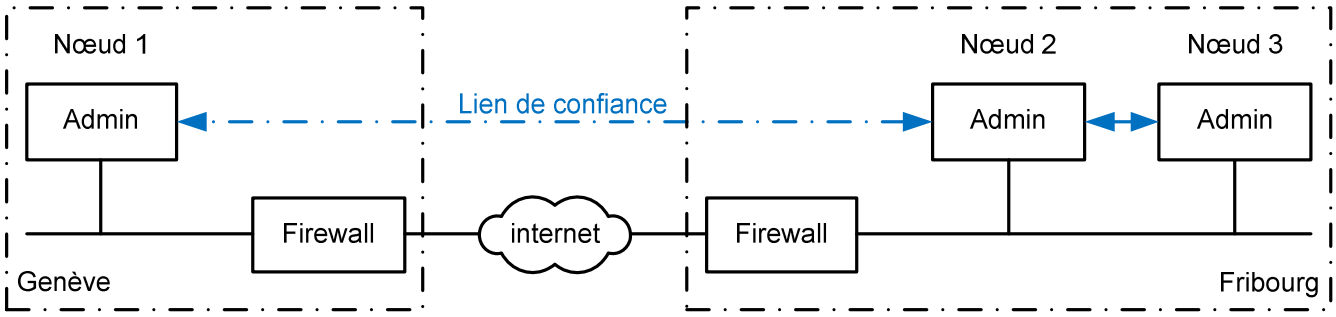
- Christian **Abegg**, étudiant Bachelor en informatique
- Sébastien **Pasche**, étudiant Master en TIC
- Cédric **Penas**, assistant Ra&D
- Lionel **Schaub**, étudiant Bachelor en informatique puis assistant Ra&D

[1] http://tic.rcso.ch/
[2] http://hepia.hesge.ch/
[3] http://www.tdeig.ch/

Partie 1 : Contexte général et donnée du problème

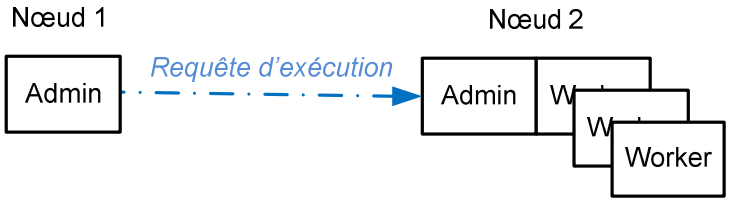
1.1 Introduction

La figure ci-dessous illustre l'architecture générale de la plateforme Visag constituée de 3 nœuds et de 2 liens de confiance. Des liens de confiance sont également présents entre les VMs Admin et leurs VMs Worker mais ne sont pas représentés ici.



L'exemple montre un nœud physique situé à Genève (hepia) et deux nœuds à Fribourg bénéficiant de la sécurité physique propre à chaque établissement.

L'échange inter-sites, assuré par les machines virtuelles Admin, utilise internet via un lien de confiance établi manuellement lors de la configuration des nœuds.



Chaque nœud (dans l'exemple ci-dessus le noeud 2) est à l'écoute d'un nœud ami (nœud 1)

Chaque nœud va ensuite démarrer une(des) machine(s) virtuelle(s) Worker en fonction du cadre applicatif. Nous utiliserons l'abréviation VM (Virtual Machine) dans la suite de ce document.

1.2 Nœud & Configuration des adresses IP

- Un **nœud** est un ordinateur de type PC compatible x86 → Voir matériel[4] utilisé à hepia
- Chaque nœud possède 1 VM Admin configurée avec une adresse IP **statique**
- Les VMs Workers sont créées dynamiquement et reçoivent une adresse IP **dynamiquement** (pool DHCP)
- Le nombre de VMs Workers dépend du contexte applicatif ; il a été choisi arbitrairement à 3 dans l'exemple
- A l'installation, 2 fichiers (VM_Admin.ovf + VM_Worker.ovf) doivent être déployés avec vSphere (disque local)

[4] http://www.tdeig.ch/vmware/Montage_PC_Gigabyte.pdf

1.3 VMs Admin & Worker

VM Admin :

- OVF disponible ici : http://valentin.clement.home.hefr.ch/visag/ovf/visag_admin_64.zip

Guest OS:	Ubuntu Linux (64-bit)
VM Version:	7
CPU:	1 vCPU
Memory:	512 MB
Memory Overhead:	153,91 MB
VMware Tools:	OK

- OS utilisé : Linux Ubuntu Server 10.10 64bits (kernel 2.6.35)
- Le niveau applicatif vPopC adresse une demande à la VM_Admin qui va communiquer avec son hyperviseur pour vérifier les ressources disponibles du serveur puis elle va donner l'ordre de cloner la VM_worker (les fichiers vmdk, …) afin de traiter la demande reçue.

VM Worker :

- OVF disponible ici : http://valentin.clement.home.hefr.ch/visag/ovf/visag_worker1.zip

Guest OS:	Ubuntu Linux (64-bit)
VM Version:	7
CPU:	1 vCPU
Memory:	512 MB
Memory Overhead:	88,83 MB
VMware Tools:	OK

- OS utilisé : Linux Ubuntu Server 10.10 64bits (kernel 2.6.35)
- La VM_Worker sert uniquement de cadre d'exécution pour les jobs d'une application. Lorsqu'elle est sollicitée par la VM_Admin, un clone est créé puis un « reverse snapshot » effectué afin de pouvoir exécuter le(s) job(s).
- Pourquoi le snapshot ?

  Le snapshot avait été implémenté par Wyssen dans la première version de vPOPC car il n'y avait à ce moment-là pas encore de Workers dynamiques. Il fallait donc assurer qu'à chaque nouvelle exécution, la VM se trouvait dans un état **sain** grâce au snapshot.

  Il permet également un gain en temps car c'est beaucoup plus rapide de faire un « reverse snapshot » plutôt que de démarrer l'OS.

1.4 Configuration des liens confiance

Un lien de confiance garantit aux 2 partenaires (endpoints) un échange sécurisé entre 2 nœuds amis

Pour créer ces 2 liens de confiance, il convient d'effectuer les opérations suivantes dans les **VMs Admin** :

Sur le nœud 1 :
- Générer la paire de clés SSH
- Entrer l'IP du nœud 2 (lors de l'installation service) en tant que nœud de confiance

Sur le nœud 2 :
- Générer la paire de clés SSH
- Entrer l'IP du nœud 1 (lors de l'installation service) en tant que nœud de confiance
- Entrer l'IP du nœud 3 (lors de l'installation service) en tant que nœud de confiance

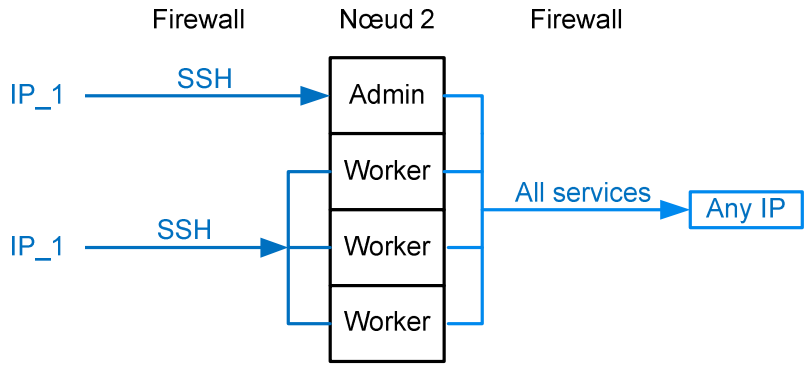
Sur le nœud 3 :
- Générer la paire de clés SSH
- Entrer l'IP du nœud 2 (lors de l'installation service) en tant que nœud de confiance

Puis inscrire sur chaque nœud (dans `.ssh/authorized_keys`) les clés publiques des nœuds avec lesquels il y a un lien de confiance.

Lors de l'établissement du canal SSH, l'authentification est dite mutuelle puisque chaque extrémité vérifie que le partenaire est le bon :
- Nœud 1 : contrôle l'authentification ← Nœud 2
- Nœud 1 → Nœud 2 : contrôle l'authentification

1.5 Flux de données à autoriser dans les firewalls

Illustration avec le nœud 2 qui doit pouvoir utiliser 2 liens de confiance:



Il faut **3 règles** pour que le nœud 2 puisse utiliser ces 2 liens de confiance :
1. Autoriser les connexions SSH (port TCP 22) entrantes destinées à la VM Admin
2. Autoriser les connexions SSH (port TCP 22) entrantes destinées aux VMs Workers (Pool d'adresses)
3. Autoriser les connexions sortantes

1.6 Scénario de test

Dans un premier temps, une analyse de flux avait été réalisée sur la première version de vPOPC qui n'utilisait pas le protocole SSH pour toutes ses connexions.
Cette analyse peut être consultée à l'adresse suivante : http://www.tdeig.ch/visag/demo/Analyse_Execution.pdf

La dernière version de vPOPC implémente des échanges protégés par le SSH.

Le scénario mis en place utilise un programme de démo (Demopopc) mis au point par Fribourg qui consiste à créer un certain nombre d'objet sur notre réseau de nœud et à leur faire échanger des ID.

Notre réseau est composé de 5 machines ESXi reparties entre les sites de Genève et Fribourg :

- Genève (1 machine) : 129.194.184.96
- Fribourg (4 machines) : 160.98.22.79, 160.98.22.86, 160.98.22.91, 160.98.22.96

Les résultats de cette analyse se trouvent dans l'annexe 4.

1.7 Redondance de la plateforme

Cette plateforme peut donc comprendre un nombre élevé de nœuds offrant ainsi une **redondance** très utile à ces utilisateurs.

Partie 2 : Analyse

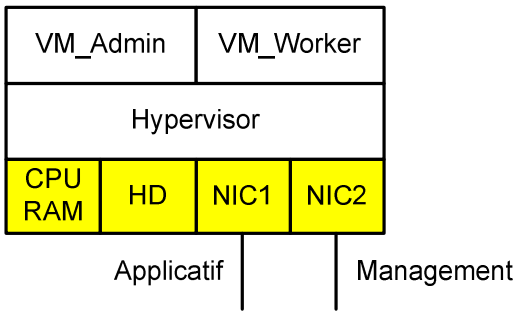
2.1 Objectifs

- L'analyse de sécurité de cette plateforme exige un référentiel basé sur les bonnes pratiques.
- Elles se veulent génériques et donc indépendantes du type d'hyperviseur utilisé.
- Nous avons choisi pour cette étude ESXi (version 4) comme hyperviseur suite à son utilisation depuis 2009 au labo
- Voir rubrique Virtualisation[5] ainsi que nos formations[6] [7] VMware suivies par plus de 50 personnes provenant des secteurs banque, santé, services & formation.

2.2 Décomposer

La problématique sécuritaire de cette plateforme est décomposée en **5 domaines principaux :**

1. **Réseau** → séparation des réseaux → défense périmétrique
2. **Système** → comptes actifs, services actifs, logs
3. **Management** → accès distant, backup, …
4. **Virtualisation** → cloisonnement des VMs, …
5. **Applicatif** → flux, …



L'hyperviseur (Hypervisor) constitue la couche logicielle qui accède au matériel (CPU, RAM, HD, NIC) et qui fait croire à chaque VM qu'elle est seule à accéder au matériel.

Tout hyperviseur (ESXi et KVM étudiés dans ce projet) doit respecter 3 principes fondamentaux :

- Compatibilité binaire (x86) avec un système physique (Operating system + applications)
- Transparence pour offrir les meilleures performances (mode Kernel, Intel VT, …)
- Cloisonnement (isolation) entre les VMs

Il n'est pas prévu d'intégrer la problématique du stockage (SAN, …) dans cette étude si bien que nous utiliserons un disque physique (HD = Hard Disk) sur chaque nœud.

2.3 Risques

2.3.1 Disponibilité des nœuds et des services virtualisés

Garantir la présence d'un service téléphonique ou informatique n'est pas une mission simple.
Des surcharges peuvent être dues à des ressources matérielles insuffisantes et/ou à une croissance non-maîtrisée des affaires.
Les solutions proposées au §3 utilisent la notion de qualité de service et des profils des systèmes

Par contre l'attaque de type Denial of Service (DoS/DDoS), consistant à bombarder la victime (votre serveur web ou le canal de confiance dans notre cas) de paquets, présente sur internet un risque important dans un environnement de production.
Certains spécialistes parlent de ranconware car la victime reçoit généralement une demande de rançon ; voir la suite de l'histoire dans l'excellente revue MISC No19[8]

[5] http://www.tdeig.ch/
[6] http://www.tdeig.ch/vmware/Flyer1.pdf
[7] http://www.tdeig.ch/vmware/Flyer2.pdf
[8] http://ed-diamond.com/produit.php?ref=misc19&id_rubrique=8&caracteristique=1-2-&caracdisp=2-9-

Nous montrerons au §3.11 comment un superviseur peut identifier ce genre de problème
Toutefois nous préférons concentrer nos efforts sur d'autres risques car celui-ci est bien connu et les solutions sont les mêmes pour une architecture virtualisée que pour une architecture physique.

Dans un environnement critique, les conséquences d'un crash doivent être analysées

- Où sont les sauvegardes ?
- Quel est le temps nécessaire pour remonter complètement un nœud physique ?

Remarquons, dans le cas particulier de cette grille de calcul qu'il n'est pas nécessaire de sauvegarder les données applicatives

La redondance présente dans cette grille de calcul (§1.7) sert évidemment à augmenter sa disponibilité.

2.3.2 Intégrité

Le contrôle d'intégrité doit déceler toute modification sur un fichier
Lors du téléchargement de la mise à jour d'un logiciel, ce contrôle (effectué par le protocole TLS/SSL) garantit que le fichier reçu correspond exactement (au bit prêt) à l'original.

L'intégrité est certainement l'élément le plus critique pour garantir un service digne de confiance (=sûr).
Idéalement je devrais être certain que le système d'exploitation qui vient d'être chargé lors de la mise sous tension de mon Windows correspond exactement à celui qui a été installé voici 18 mois à partir du CD.
La puce TPM de votre portable va effectuer ce contrôle.

L'étude du mode de fonctionnement de cette plateforme via les flux échangés permet d'identifier plusieurs risques :

1. Chaque nœud doit exécuter du code légitime (hyperviseur + VMs)
2. Chaque nœud doit garantir l'intégrité de calcul (la VM1 hostile ne doit pas pouvoir modifier le calcul effectué par la VM2 sur le même noeud)

Voir §2.4 pour son analyse spécifique dans une architecture virtualisée

2.3.3 Confidentialité

La confidentialité consiste à modifier (= chiffrer) les données pour que l'ennemi ne puisse pas les utiliser

Voici le même texte chiffré en effectuant une rotation de 13 caractères
Yn pbasvqragvnyvgé pbafvfgr à zbqvsvre (= puvssere) yrf qbaaérf cbhe dhr y'raarzv ar chvffr cnf yrf hgvyvfre

Voici le même texte chiffré ???

```
EF 15 6C 3E F4 DC F6 50 FC 99 2F 56 9A 04 2B 2F 77 4F 88 54 9A BC D4 5E EC F9 88 0E 27 2F AF 83 8D D9
DD 3E F7 11 87 DE E2 82 FE EC AB 90 47 E5 79 C9 9E A6 1B 2C CB 14 8E 2E FE 15 8B C2 25 44 CB 54 4A 34
D0 52 34 08 AE D1 38 E3 CD 33 33 BF 0F E1 11 68 10 20 A4 72 E4 FF 66 77 60 AD 4B 5A 3F 03 55 C8 2A 31
9D 67 C5 4D 98 9D 09 97 CA 1A
```

Dans notre cas, les résultats du calcul effectué dans une VM Worker circulent chiffrés (SSH) dans la grille

Les flux d'administration sont chiffrés.

2.4 Virtualisation

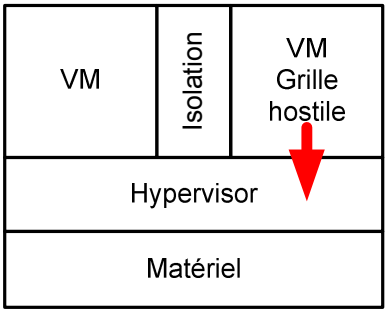
2.4.1 Des questions légitimes sont fréquemment débattues par des utilisateurs ou des spécialistes de la virtualisation :

- La virtualisation a-t-elle un impact négatif sur la sécurité de cette plateforme ?
- Peut-on auditer une architecture virtualisée selon les principes utilisés pour le monde physique ?
- La virtualisation peut-elle augmenter la sécurité ?

2.4.2 Les principaux risques de sécurité

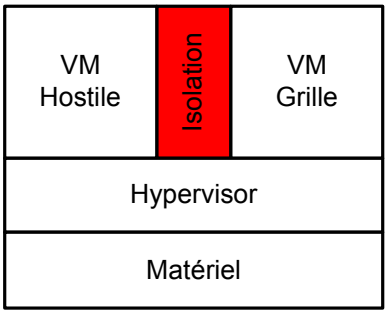
Danny Corriveau[9] présente une analyse particulièrement illustrative des risques potentiels

- Slide 10 : les risques de la VM sont analogues à ceux du monde physique
- Slide 11 : Avec ou sans virtualisation, la problématique est la même
- Slide 12 : La VM tente d'exploiter des failles via l'interface de programmation qui n'est pas publié dans le cas d'ESXi → voir publication de Ken Kato[10]
  Cette attaque est désignée par → VM Escape



A l'inverse, la solution KVM s'appuie sur des appels système publiés

- Slide 13 : Si l'isolation est garantie entre les VM, la VM Hostile ne peut pas accéder (lire ou écrire) à la VM Grille ; ainsi les calculs effectués sur une VM Grille ne peuvent pas être altérés par une application tierce (malveillante = hostile) s'exécutant dans une VM voisine garantissant donc l'intégrité du calcul applicatif de la grille
  Cette attaque est désignée par → VM Hopping



- Slide 14 : Attaque de l'hyperviseur depuis la couche physique
  Le risque existe qu'une faille au niveau gestion mémoire ou dans la gestion des communications puisse être exploitée
- Slide 15 : Attaque de la VM depuis l'hyperviseur
- Slide 16 : Risque réel qu'un espion puisse lire la mémoire
  Voir §5.6 Empirical Exploitation of Live Virtual Machine Migration
  Voir §3.0 Conseils en matière d'architecture virtualisée

[9] http://www.tdeig.ch/visag/Danny_Corriveau_2008.pdf
[10] http://www.tdeig.ch/visag/Backdoor.pdf/

2.4.3 Objectifs de cette étude

Les recherches sur internet (voir §5.7) confirment que l'hyperviseur ESXi ne présente pas de risque majeur au niveau isolation bien qu'aucune preuve formelle ne puisse être démontrée.

Nos connaissances théoriques sont résumées au § 5.11 VMM : la clé de voute de la sécurité
Dans une architecture de type Hypervisor, le composant VMM (Virtual Machine Monitor) émule la couche matérielle (CPU, RAM, …) et garantit que la VM ne puisse accéder qu'à son propre espace mémoire physique.

Nous profiterons de ce financement pour mieux comprendre le cadre d'exécution d'une VM :

- **Démontrer qu'une VM ne contient pas nécessairement un système d'exploitation**
  Christian Abegg a déniché la base logicielle capable de charger les instructions x86 du programme à exécuter
  Voir son rapport[11], sa présentation[12] et les fichiers utiles[13]

- **Déterminer dans quel mode (Ring) la VM démarre son exécution**

- **Démontrer que l'espace mémoire RAM est limité à la valeur Memory_Size définie lors de la création de la VM**
  Ce mécanisme, implémenté au niveau du VMM, est essentiel pour garantir l'isolation
  Y a-t-il des traces dans les logs ?

- **Démontrer que l'espace RAM disponible est initialisé**
  Une VM qui vient de démarrer ne doit pas pouvoir lire des données issues d'une autre VM qui aurait utiliser précédemment ce même espace physique

- **Changer de mode par exemple de Ring0 à Ring3**

- **Répéter ces tests avec l'hyperviseur Linux-KVM**
  Les résultats devraient être similaires

2.4.4 Résultats

Rapport de l'étude bas niveau d'une VM grâce au projet SOS
http://www.tdeig.ch/visag/sos/SOS_Rapport.pdf

VM contenant le code source SOS modifié + script de déploiement de l'image
http://www.tdeig.ch/visag/sos/ova/Ubuntu.ova

VM SOS permettant d'exécuter le code produit
http://www.tdeig.ch/visag/sos/ova/SOS.ova

Code source SOS modifié
http://www.tdeig.ch/visag/sos/SOS_penas.tar.gz

2.4.5 Orientation des futurs travaux d'étudiants

Peut-on analyser sérieusement la sécurité d'un logiciel lorsque l'on n'a pas accès au code source ?
La méthodologie consiste à faire subir une série de tests à la boîte noire.
Selon une méthode black-list, nous devons identifier tous les cas de test permettant de créer une situation anormale.
Nous comprenons vite les limites de la démarche

[11] http://www.tdeig.ch/vmware/Abegg_R.pdf
[12] http://www.tdeig.ch/vmware/Abegg_P.pdf
[13] http://www.tdeig.ch/vmware/Abegg/

Security by obscurity
Les experts de la sécurité préfèrent un algorithme de chiffrement ou une fonction de hachage publiée plutôt qu'une solution tenue secrète.
Le cas du produit SecurID me semble en être un parfait exemple :

1. Le token est présenté comme une géniale invention tenue secrète capable de générer 6 chiffres de manière aléatoire à chaque minute
2. Des curieux cherchent et finissent par en comprendre l'algorithme
3. Des logiciels comme Cain[14] permettent de remplacer (émuler) un jeton oublié ou perdu !

Time Hash	Local Time
048979	2011/11/30 - 10:37
197368	2011/11/30 - 10:38
464153	2011/11/30 - 10:39
494549	2011/11/30 - 10:40
152169	2011/11/30 - 10:41
771078	2011/11/30 - 10:42
923648	2011/11/30 - 10:43
793429	2011/11/30 - 10:44
810494	2011/11/30 - 10:45
553683	2011/11/30 - 10:46

Avec une sécurité basée sur l'obscurité il est impossible de valider scientifiquement la sécurité ESXi sauf en entrant dans l'illégalité avec une démarche de type *reverse engineering*

VMsafe comme outil de développement
Il me semblait intéressant en mars 2010 d'entrer dans le programme **VMsafe** pour proposer à nos étudiants des travaux associés à la sécurisation d'une plateforme ESXi en liaison avec des risques majeurs (contrôle d'intégrité) que nous avions identifiés.
La réponse de VMware *"Unfortunately VMsafe is not part of the Academic program"* a été une des raisons de migrer sur la solution Open Source Linux KVM[15]
La formation Advanced VMware Security – Ultimate Bootcamp – Lancelot, que j'ai suivie en oct 2010, confirme le côté fermé de VMsafe pour le monde académique.
Voir aussi §5.4, §5.5, §5.8 et §5.9

Est-il facile de valider la sécurité d'un hyperviseur Open Source ?
Beaucoup de personnes vantent la démarche mais peu d'individus ont le courage, le temps et les connaissances pour arriver au but !
Les résultats (comme Live Replication of Paravirtual Machines[16]) sont plus riches si les conditions précédentes sont réunies
Le lecteur curieux pourra parcourir l'étude[17] [18] [19] de Sébastien Pasche effectuée dans le cadre de son Master

Complexité
La virtualisation est passionnante et complexe ; ses limites sont celles de tout individu
Quelques liens[20] [21] qui dépassent hélas le cadre de cette étude.
Voir aussi §5.12

Perspectives pour notre économie : critiquer les prix pratiqués ou utiliser une solution Open Source ?
La solution ESX-ESXi est largement utilisée depuis des années car VMware a été le premier à vendre un hyperviseur sûr, performant et stable.
Pour des besoins modestes (4 VMs) comme ceux du labo, la solution gratuite ESXi semble idéale.
Par contre le coût actuel (2011) des licences est critiqué par une large majorité des entreprises qui ont besoin du vCenter, de vMotion, de HA, …

[14] http://www.oxid.it/
[15] http://www.redhat.com/f/pdf/rhev/DOC-KVM.pdf
[16] http://deposit.ddb.de/cgi-bin/dokserv?idn=996373381&dok_var=d1&dok_ext=pdf&filename=996373381.pdf
[17] http://www.tdeig.ch/kvm/pasche_R.pdf
[18] http://www.tdeig.ch/kvm/pasche_R.pdf
[19] http://www.tdeig.ch/kvm/pasche/
[20] http://www.coseinc.com/en/index.php?rt=download&act=publication&file=Introducing%20Blue%20Pill.ppt.pdf
[21] http://www.eecs.umich.edu/virtual/papers/king06.pdf

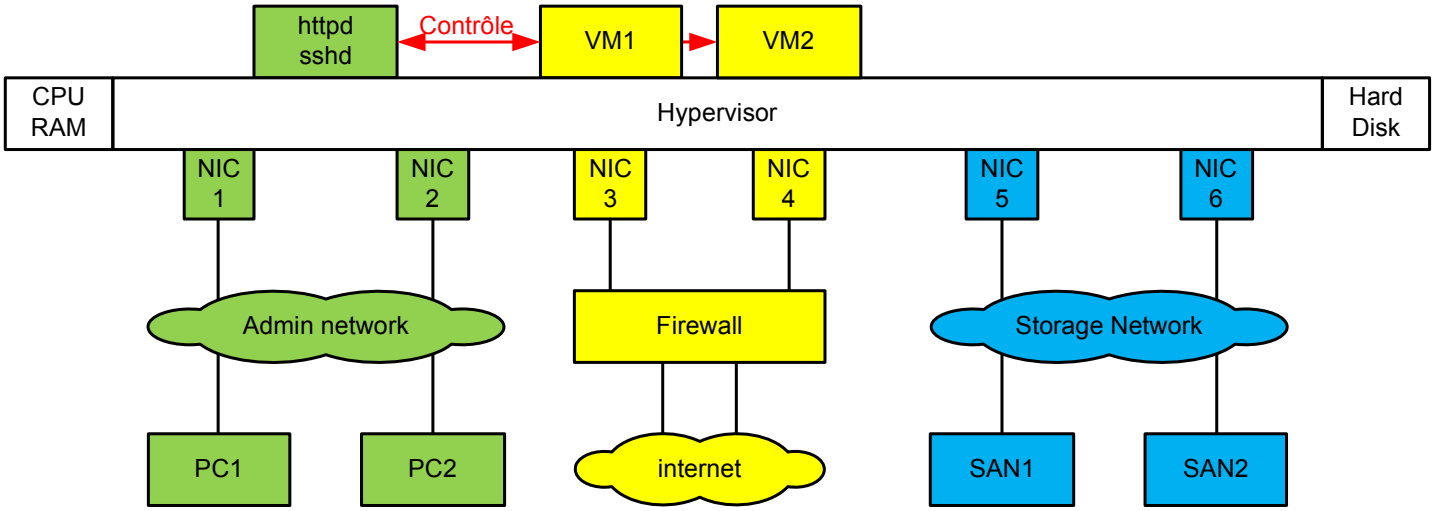
Conclusion

La sécurité repose sur un lien de confiance.
Avez-vous confiance dans le nouveau système d'authentification que vous venez d'installer ?
Faute de temps et parfois de compétences, l'utilisateur final doit souvent faire confiance aux produits utilisés.
Les écoles HES peuvent être le partenaire pour aider une entreprise à faire les bons choix ou pour valider (auditer, tester) une infrastructure existante.
Les résultats sont souvent convaincants car le risque majeur demeure au niveau de l'être humain … qui a oublié …qui a mal configuré, qui n'a pas compris que …

3.0 Mes conseils en matière d'architecture virtualisée

Identifier les principaux risques
Construire une défense en profondeur
Privilégier la sécurité comportementale basé sur un modèle liste blanche
Définir des zones en fonction du contexte applicatif et des risques associés
Choisir des briques de confiance → trusted computing
Admettre que tout dispositif de sécurité a ses limites (un faux négatif correspond à une intrusion non détectée)

Nous proposons ce modèle pour illustrer la problématique :



- L'hyperviseur est le composant central responsable de **l'isolation** au niveau des processeurs (CPU), de l'espace mémoire RAM et des réseaux (vert-jaune-bleu).
  En matière de confiance, je la qualifie d'aveugle puisque les sources de ESX-ESXi ne sont pas disponibles. Elle repose donc sur la capacité de VMware à faire évoluer son produit phare tout en garantissant à ses utilisateurs une isolation parfaite.

- La partie verte illustre l'administration depuis 2 PCs qui communiquent avec les processus httpd ou sshd présents sur tout ESX-ESXi.
  Elle doit être considérée comme **digne de confiance** y compris au niveau des **êtres humains** qui gèrent cette infrastructure.
  Dans l'idée du sas de sécurité, la personne chargée de l'administration ne disposera d'aucune connexion internet et ne branchera aucune clé USB dans le réseau vert, … Voir §3.3 & 3.5.4.5
  Il est possible de faciliter ce travail d'administration via des outils spécifiques comme le superviseur développé par Lionel Schaub ; voir au §3.11

- La partie bleue n'a pas été traitée dans cette étude afin d'en limiter le coût.
  Le lecteur intéressé peut trouver quelques conseils dans les études suivantes effectuées par Benoit-Georges Chalut[22] [23] , Cédric Penas[24] [25] et Loïc Sandmeier[26] [27]
  Cette partie est vitale en matière de disponibilité ; voir §3.9
  Elle doit être considérée comme **digne de confiance** y compris au niveau des **êtres humains** qui gèrent cette infrastructure ; voir §3.3

- La partie jaune est la plus intéressante (critique) car elle comprend les différentes applications proposées et intégrées dans des machines virtuelles.
  L'illustration décrit 2 machines virtuelles (VM1 & VM2) qui offrent leurs services via internet.
  Elle offre donc des niveaux de risque variables qui dépendent du contexte applicatif.
  Les bonnes pratiques sécuritaires du monde physique restent heureusement efficaces.

[22] http://www.tdeig.ch/vmware/Chalut_P.pdf
[23] http://www.tdeig.ch/vmware/Chalut_R.pdf
[24] http://www.tdeig.ch/vmware/penas_M.pdf
[25] http://www.tdeig.ch/vmware/penas_P.pdf
[26] http://www.tdeig.ch/vmware/sandmeier_M.pdf
[27] http://www.tdeig.ch/vmware/sandmeier_P.ppt

3.1 Documentation & simplicité

Documentation

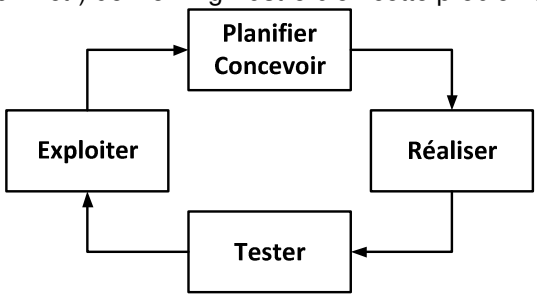
- Chaque être humain possède des limites physiques (capacité de mémorisation, …)
- Les employés (mes assistants) vont et viennent ; il faut donc capitaliser leur savoir
- Il est crucial de disposer d'une documentation appropriée (qui doit être ni trop légère car inutile, ni trop riche car trop chère)
- Elle doit respecter les principes d'attributs de la qualité de l'information
  - Le contenu doit être exact, exhaustif, concis, …
  - L'information doit être actuelle, à jour … → Penser à la durée de vie de cette information
  - Sa forme (présentation) doit être claire, structurée, …
- Mon PC actuel possède 172'000 fichiers ; un grand merci à Goggle Desktop qui facilite chaque jour mon travail

Simplicité

- Pourquoi utiliser des usines à gaz pour des besoins simples comme l'échange d'email ?
- Les systèmes d'information deviennent toujours plus complexes
- La virtualisation ajoute une complexité supplémentaire (VMware a enfin accepté l'idée)
- On ne maîtrise bien que ce que l'on comprend (lapalissade)
- Plus un système est simple; plus il est facile à sécuriser ; à l'inverse plus le système est complexe plus mon ennemi (hacker) …

3.2 Gestion du changement *(change management)*

- L'Homme n'aime pas le changement
- Des mécanismes puissants prônent l'innovation, la dernière technologie, le risque d'obsolescence, …
- Je dois avouer que la documentation de l'infrastructure (LAN, firewall, serveurs DNS-web-SSL-PKI, …) de mon labo reste parfois lacunaire et peine à suivre les changements. Heureusement que les risques sont jugés faibles avec des *assets* pédagogiques
- Le cycle PDCA (Plan-Do-Check-Act-) de Deming illustre bien cette problématique



- En entreprise, comment garantir que le niveau de sécurité du nouveau serveur, qui vient de passer les tests de validation pour être mis en production, reste au même niveau ces 3 prochaines années ?
- Penser à systématiser les tests (unitaires, …, fonctionnels) car beaucoup d'anomalies ou de démonstrations foireuses sont dues à des tests insuffisants !

3.3 Sécurité physique

- Sans elle, la sécurité logique devient inutile
- Elle est donc indispensable à l'image de l'étourdi qui oublie de fermer la porte de son appartement
- Elle simplifie grandement le travail du RSSI face à des gourous capables de lire une clé privée à partir de signaux électriques mesurés dans un serveur, face aux risques d'évasion des données stockées dans le SAN, face aux risques de connexion dans le réseau d'administration, …
- Elle semble presque oubliée dans les messages publicitaires vantant un Cloud parfait : personne ne sait à qui il appartient, où il est et comment il fonctionne
- J'espère que les environnements critiques (centrale nucléaire, …) sont bien isolés d'internet
- Le risque de mauvaise configuration demeure si les mêmes équipements physiques (routeur, …) d'un opérateur (Internet Service Provider) offrent simultanément des services privés (Intranet, Virtual Private Network, …) et publics (Internet)
- Le risque d'intrusion demeure comme le malware stuxnet l'a montré récemment

3.4 Administration

3.4.1 Le cockpit des avions de ligne est aujourd'hui fermé à clé
Il en va de même pour celui d'un centre informatique
Le flux de management reste confiné au bâtiment → sécurité physique du §3.3

3.4.2 Ce flux est protégé par un protocole sécurisé (SSL ou SSH) assurant :
1. Authentification du nœud (très important)
2. Intégrité des données échangées (très important)
3. Confidentialité des données échangées (défense en profondeur puisque le réseau est séparé physiquement)

3.4.3 Une fois le contrôle d'accès résolu, il convient de réfléchir au modèle organisationnel de l'entreprise
- Qui possède les droits suprêmes ?
- Pourquoi ?
- Avons-nous défini des rôles possédant des vues limitées ?
- Qui gère les serveurs applicatifs ?
- Qui gère le réseau et la défense périmétrique ?
- Qui a une vue d'ensemble ?
- …

3.4.4 Remarquons que l'administration à distance de certains serveurs (web, …) peut conduire à des risques élevés en matière d'image de marque → web defacing

3.5 Systèmes

3.5.1 Utiliser, grâce au protocole ntp une seule référence temporelle pour l'hyperviseur & les VMs
Important pour les logs → voir §3.10.8

3.5.2 Hyperviseur
Préciser la version utilisée par exemple ESXi 4.0.0 Build 208167
Sauvegarder les fichiers de configuration
Identifier son **profil** → comptes créés, fichiers de config, …

3.5.3 Documenter chaque VM légitime (= autorisée)
Connaître son **profil** qui va devenir une référence → Voir §3.10.4
Comptes créés sur la VM_Admin et sur chaque VM_Worker
Droits des systèmes de fichiers
Relevé de mesures (min – moyen – max) des paramètres CPU – RAM – Storage – Network
Le compléter éventuellement avec des mesures du Guest OS
Penser à le suivre périodiquement ; à le mettre à jour : 10 utilisateurs au début – 20 aujourd'hui – X demain

3.5.4 Mesures classiques à appliquer pour l'hyperviseur
3.5.4.1 Effectuer un contrôle d'intégrité lors de l'installation
3.5.4.2 Configurer et tester une authentification mutuelle pour les postes d'administration → Voir 2.9.4
3.5.4.3 Définir des comptes spécifiques appropriés au contexte applicatif (business)
3.5.4.4 Désactiver les pilotes et les services inutiles
3.5.4.5 Supprimer les ports physiques (USB, …) et périphériques inutiles
3.5.4.6 Considérer une VM comme une machine physique (anti-virus, …)
3.5.4.7 Activer les logs pour les envoyer sur un serveur syslogd
3.5.4.8 Activer des mécanismes de type white list (SELinux, …), procéder à du hardening
3.5.4.9 Définir quand et qui applique les correctifs, updates, patches, …
3.5.4.10 Ne pas avoir une confiance aveugle pour la sécurité par l'obscurité : cas SecurID, … !!!

3.5.5 Mesures classiques à appliquer pour chaque VM
3.5.5.x Comme ci-dessus

3.6 Redondance

- La mise en parallèle de ressources est très fréquente pour augmenter la disponibilité et/ou augmenter les performances
- Distinguer les mécanismes exclusifs *Hot swapping* (firewall1 ou firewall2) des mécanismes *Load Balancing* (N serveurs web accessibles simultanément)

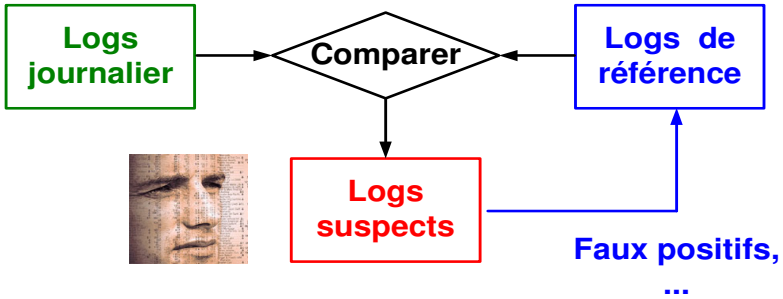
3.7 Réseau

3.7.1 Il est conseillé de séparer physiquement les flux d'un nœud en utilisant des cartes réseau
Les cartes (NIC = Network Interface Card) récentes supportent un débit binaire de 1 Gbit/s
1 carte au minimum par réseau ; 2 dans notre cas (Voir figures du §2.2 et 3.0)
Une séparation classique de type VLAN peut être envisagée

3.7.2 Configurer les règles du firewall à partir du §1.5 (défense périmétrique)
Seuls les flux applicatifs des machines virtuelles VM_Admin et VM_Worker traversent le firewall et internet.

3.7.3 Tester les ports (TCP/UDP) ouverts

3.7.4 Savez-vous ce qui circule sur vos réseaux ? Volume ? Trafic légitime ? …
Ces métriques servent à définir la qualité de service → Service Level Agreement (SLA)
Très souvent, la présence de malwares peut être observée avec un sniffer (outil capable d'analyser les paquets circulants sur le réseau) car ils (keylogger, …) doivent communiquer avec leur maître
Le principe de base consiste à comparer le trafic journalier par rapport à un profil jugé normal et construit dans un environnement sure



3.8 Applicatif

Le contexte applicatif a été introduit au §1

3.8.1 Lien de confiance
Chaque nœud doit être certain de communiquer, en tout temps, avec un nœud ami.
Choix = tunnels SSH
Les paramètres suivants sont introduits manuellement sur les 2 nœuds (du lien de confiance) :

- Adresse IP du nœud partenaire
- Clé publique du nœud partenaire

3.8.2 Tester la bonne configuration du protocole SSH
3.8.2.1 Fichiers de configuration
3.8.2.2 Génération d'une paire de clés
3.8.2.3 Clé de session
3.8.2.4 Bonnes pratiques de configuration d'un serveur
3.8.2.5 Test de pénétration depuis internet

3.8.3 Quel type de *guest OS* (Linux, Windows, …) est-il prévu d'utiliser ?

3.9 Stockage & Sauvegarde

3.9.1 Définir les biens à protéger, les risques, l'impact financier et les SLAs

3.9.2 Effectuer une sauvegarde des fichiers de configuration de l'hyperviseur et des VMs

3.9.3 Effectuer une sauvegarde régulière de chaque VM

3.9.4 Tester la capacité à restaurer un système ou une VM dans le temps convenu selon le SLA

3.9.5 Continuer de sauvegarder les données applicatives selon une méthode classique

3.9.6 Les machines virtuelles sont stockées sur le disque local et sauvegardées sur un serveur de fichiers externe accessible dans le réseau de management via les protocoles iSCSI ou SMB.

3.10 Supervision

3.10.1 Assurer que le niveau de sécurité de la plateforme reste le même dans la durée !!!

3.10.2 Comprendre les enjeux et les méthodologies

3.10.3 Eviter les comportements naïfs car l'adversaire est souvent malin

3.10.4 Disposer de références → profils systèmes et réseau

3.10.5 Surveiller l'occupation des ressources physiques → voir §3.11

3.10.6 Surveiller les performances (qualité de service) afin de les comparer au SLA

3.10.7 Audit régulière (fichiers de config, …) pour détecter des anomalies comme rogue VM = VM non autorisée

3.10.8 Analyser (comprendre) les logs, contrôler la taille des logs, effectuer des sauvegardes

3.11 Travail de Bachelor de Lionel Schaub : Superviser un Cloud Computing

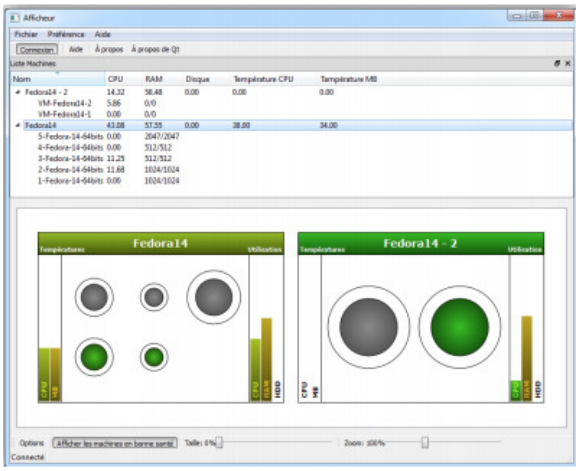
Extrait de l'énoncé[28]
L'objectif de ce travail consiste à développer un superviseur capable d'afficher l'état de santé des (10-100-1000) serveurs qui composent cette grille (Cloud Computing) et qui offrent un service virtualisé.
Ce logiciel, sur le modèle du cockpit de l'avion, renseignera sur la charge des ressources partagées (CPURAM-…) et les alarmes critiques actives

Ce superviseur avait pour objectif de construire un Proof of Concept dans le temps disponible des 8 semaines disponibles

L'utilisation du Framework Qt[29] offre une grande souplesse pour générer des exécutables sur les plateformes Linux, Windows et Mac OS X.

Nous comptons poursuivre les recherches notamment avec la librairie libvirt[30]

L'auteur de ce rapport se tient à disposition de tout partenaire économique intéressé par ce produit



Présentation[31] du 30 juin 2011 lors de la défense devant le Jury

Mémoire[32] rendu le 27 juin 2011

Annexes & Sources[33]

Lionel Schaub a obtenu le note maximum de 6 avec félicitations du Jury pour son œuvre
Voici les principaux apports pour le laboratoire :

- Outils Linux pour mesurer les paramètres système tels que charge CPU, occupation RAM, performances disque, … → chap 6
- Méthodologie de mesure → chap 7 & 8
- Scénarios que je peux utiliser comme démo dans mon cours → chap 9

[28] http://www.tdeig.ch/kvm/schaub_E.pdf
[29] http://qt.nokia.com/
[30] http://www.libvirt.org/
[31] http://www.tdeig.ch/kvm/schaub_P.pdf
[32] http://www.tdeig.ch/kvm/schaub_M.pdf
[33] http://www.tdeig.ch/kvm/schaub/

Partie 4 : Analyse de sécurité de cette plateforme

4.1 Objectifs

Dans une démarche top-down, cette analyse prend en compte les spécificités des systèmes utilisés dans ce projet

Elle utilise le **référentiel présenté dans la partie 3** en évitant toute redondance inutile

La numérotation utilisée dans la partie 4 reprend celle de la partie 3.

4.2 Jugement du produit ESXi 4.0

Nos travaux relatifs à la solution gratuite ESXi a démarré en 2009 ; d'abord avec la version 3.5 puis avec la version 4.0

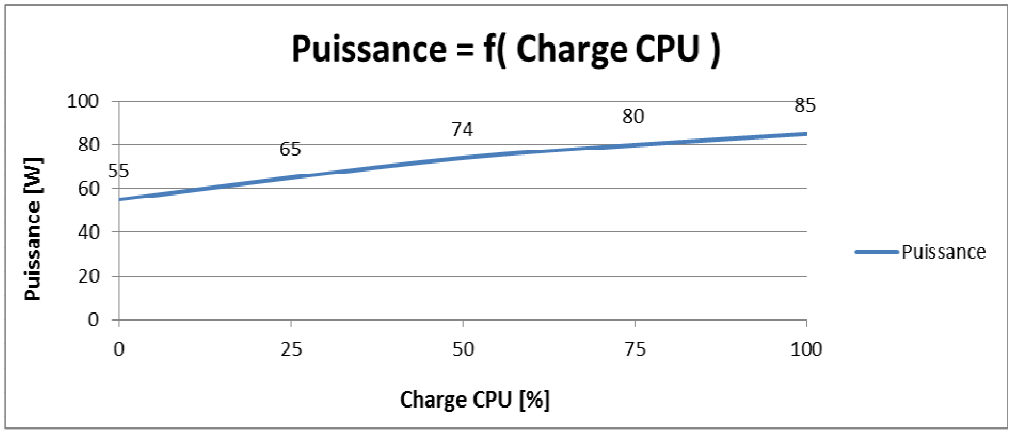
La configuration matérielle des PCs utilisés est la suivante :

- Carte mère ASUS P5Q-VM DO uATX (Intel Q45/ICH10DO - Socket 775 - FSB 1333)
- CPU Intel Core 2 Duo E8400 3.0GHz
- RAM 4Go DDR2
- Chpiset Intel® Q45 / ICH10DO support Intel Active Management Technology
- LAN Intel 82567LM Gigabit LAN Phy controller

Notre avis sur le produit ESXi :

- Gratuité du produit
- Exigence matérielle peu contraignante (nos PCs coûtent moins de 1 kF) mis à part les contrôleurs Ethernet
- Facilité de mise en œuvre (installation & configuration avec l'outil vSphere)
- Mécanisme P2V (Physical To Virtual) fonctionne (Windows & Linux) mais il est préférable de l'éviter afin que les pilotes d'une VM puisse être choisi lors de la création de cette VM
- Excellente stabilité du serveur de production hébergeant 4 VMs (serveurs DNS – web – SSL – PKI) confirmée avec notre superviseur Nagios
- Dernier redémarrage complet nécessaire après avoir voulu augmenter de 1 à 2 le nombre de vCPU d'une VM Windows Server 2008 (Process ID dans esxtop avait une valeur erronée)
- Eviter d'utiliser toutes les ressources CPU → aucune réponse à ma question[34] du 4 oct 2010
- Nous n'avons pas eu besoin des logs (heureusement car ils demeurent incompréhensibles)
- **A ce jour sur la base de diverses analyses trouvées sur internet, cet hyperviseur peut être considéré comme digne de confiance en matière d'isolation (voir §2.4)**

Consommation de notre PC



[34] http://communities.vmware.com/thread/287443

4.3 Sécurité physique

Chaque nœud est sous la responsabilité d'une entité administrative (université, institut, …) qui va administrer son(ses) nœud(s) selon la politique de sécurité en vigueur actuellement pour ses serveurs : local serveurs fermé à clé.

4.4 Administration

4.4.1 Irréaliste dans un environnement école de test
La séparation des flux est présente, l'administration est limitée à l'intranet.

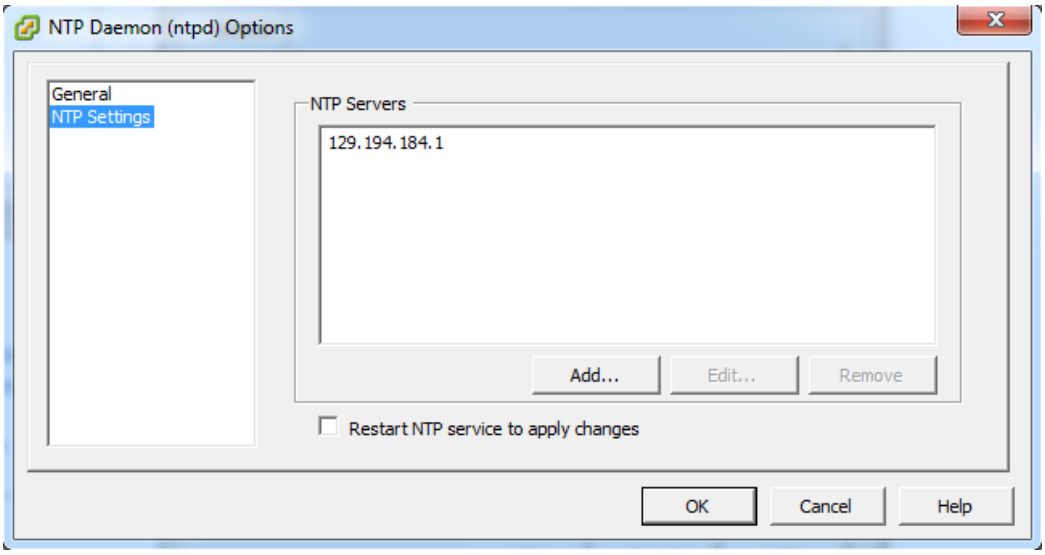
4.4.2 Ce flux est protégé par le protocole sécurisé SSL/TLS
La méthode utilisée correspond aux bonnes pratiques[35] .

4.4.3 Irréaliste dans un environnement école de test

4.4.4 Aucune administration à distance

4.5 Systèmes

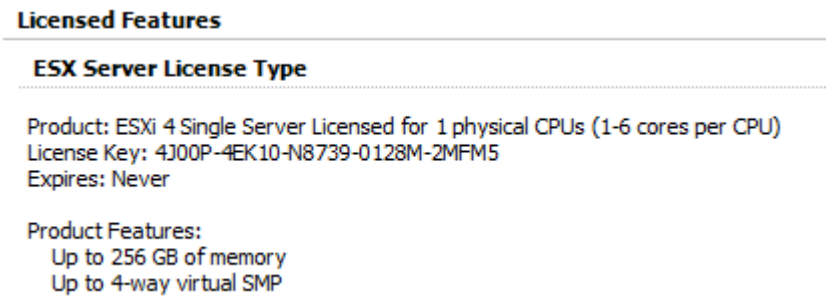
4.5.1 Procédure pour activer le protocole ntp
Sous vSphere Client, se positionner au niveau de l'hyperviseur.
Configuration -> Time Configuration -> Properties -> Options -> NTP Settings



4.5.2 Hyperviseur

Version utilisée: ESXi 4.0.0 Build 208167

Licence gratuite installée sans quoi le serveur se bloque après 60 jours.
Pour l'obtenir, ouvrir un compte sur la plateforme de téléchargement gratuite de VMware[36].
La licence doit être entrée dans vSphere Client sous Configuration → Licensed Features



[35] http://www.tdeig.ch/SSL_PKI_CA/Flyer_SSL.pdf
[36] https://www.vmware.com/tryvmware/p/activate.php?p=free-esxi&lp=default

Nous utilisons le compte root, créé lors de l'installation, pour des raisons de simplicité.
L'annexe 3 donne la marche à suivre pour créer un compte utilisateur avec des droits restreints.

Le serveur SSH disponible sous ESXi n'est pas actif par défaut.
Pour l'activer depuis la console cachée : Voir objectif 2 du document « Trucs&Astuces »[37]

4.5.3 Sur chaque Guest OS (VM), le compte **vPOCP** a été créé avec Login : visag Password : visag

Les autres comptes présents sont ceux qui ont été créés par le système Linux.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
sshd:x:102:65534::/var/run/sshd:/usr/sbin/nologin
visag:x:1000:1000:visag,,,:/home/visag:/bin/bash
```

Droits des dossiers spécifiques

En complément : http://doc.ubuntu-fr.org/securite

4.5.4.1 Le calcul du hash MD5 sur les fichiers OVF (templates fournis par Fribourg) permet d'effectuer un contrôle d'intégrité.

Logiciel utilisé : MST MD5 (outil gratuit pour windows).

VM_Worker.zip - hash md5: b1193f947c70dc91371adb27cc401dee
VM_Admin.zip - hash md5: 84d7e7c4f787760e4cca6d3c58dd6de4

Toujours mettre à disposition le md5 à côté des fichiers à télécharger.

4.5.4.2 Configurer et tester une authentification mutuelle pour les postes d'administration → Voir 2.9.4

4.5.4.3 Créer un compte ESXi avec des droits restreints → Annexe 3

4.5.4.4 Désactiver les pilotes et les services inutiles

4.5.4.5 Limiter le matériel émulé en fonction des besoins
Clic droit sur la VM → Edit settings → Onglet « Hardware »

[37] http://www.tdeig.ch/vmware/TrucsAstuces.pdf

Hardware	Summary
Memory	512 MB
CPUs	1
Video card	Video card
VMCI device	Restricted
~~Floppy drive 1~~	~~Floppy 1~~
~~CD/DVD Drive 1~~	~~CD-ROM 1~~
Network adapter 1	DMZ
SCSI controller 0	LSI Logic Parallel
Hard disk 1	Virtual Disk

4.5.4.6 Nous n'avons pas jugé utile d'installer un anti-virus sur nos VMs Linux.

4.5.4.7 Activer les logs pour les envoyer sur un serveur syslogd

4.5.4.8 Pas possible avec VMware alors qu'il semble possible de le faire sur l'hyperviseur Linux KVM

4.5.4.9 Tester avec précautions les mises à jour de l'hyperviseur car aucune assurance ne peut être fournie sur la compatibilité des versions >4.0 avec vPOPC.

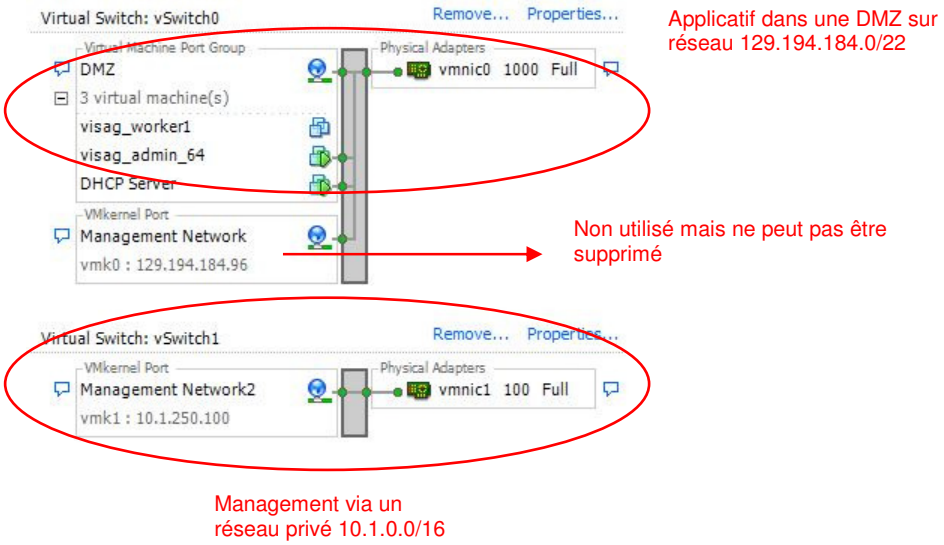
Selon la doc Ubuntu sur la sécurité[38], il est recommandé de mettre à jour l'OS régulièrement pour garder un niveau acceptable.

4.6 La redondance décrite au §3.6 n'a pas été mis en œuvre.

4.7 Réseau

4.7.1 Séparation physique du réseau d'administration et du réseau applicatif → 2 NIC nécessaires par noeud
NIC0 utilisé pour le réseau applicatif
NIC1 utilisé pour le réseau de management
Les produits Intel[39] sont compatibles ESXi

Exemple :



Le réseau applicatif doit se trouver obligatoirement rattaché au vSwitch0 car les VM_Workers qui sont créées à la volée, seront placées automatiquement sur celui-ci.

4.7.2 Règles du firewall à partir du §1.5
Les nœuds sont protégées par des *firewalls* de type *port filtering* qui vont créer les DMZs nécessaires à ce projet.

[38] http://doc.ubuntu-fr.org/securite
[39] http://www.intel.com/content/www/us/en/ethernet-controllers/ethernet-server-and-desktop-pci-x-selection-guide.html

Chaque entité d'administration (Genève - Fribourg) est libre du type de déploiement : *firewall* spécifique au projet ou ajout de règles à un *firewall* existant.

3 règles par lien de confiance sont nécessaires

1. SSH entrant sur VM Admin

Name	Action	Src If	Src Net	Dest If	Dest Net	Service
Fribourg_to_VMAdmin	*Allow*	*Internet_Dual*	*Fribourg-net*	*DMZ_If3*	*Admin_A11*	*ssh-in*

2. SSH entrant sur Workers

Name	Action	Src If	Src Net	Dest If	Dest Net	Service
Fribourg_to_Workers	*Allow*	*Internet_Dual*	*Fribourg-net*	*DMZ_If3*	*Workers*	*ssh-in*

3. SSH sortant (Tous les services dans notre cas)

Name	Action	Src If	Src Net	Dest If	Dest Net	Service
DMZ_to_internet	*Allow*	*DMZ_If3*	*If3_net*	*Internet_Dual*	*All_nets*	*All_services*

4.7.3.1 Hyperviseur accessible via l'interface NIC0
PC de test (nmap) relié à internet

```
> nmap -sUS 129.194.184.96
```

```
Starting Nmap 5.21 ( http://nmap.org ) at 2011-10-06 14:41 CEST
Nmap scan report for 129.194.184.96
Host is up (0.0011s latency).
All 1000 scanned ports on 129.194.184.96 are filtered
MAC Address: E0:CB:4E:25:2F:B2 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 21.37 seconds
```

Aucun accès. Tout est filtré par le firewall.

4.7.3.2 Hyperviseur accessible via l'interface NIC1 (réseau de management)
PC de test (nmap) relié à l'intranet

```
> nmap -sUS 10.1.250.100
```

```
Starting Nmap 5.21 ( http://nmap.org ) at 2011-09-21 16:41 CEST
Nmap scan report for 10.1.250.100
Host is up (0.00085s latency).
Not shown: 1991 closed ports
PORT     STATE         SERVICE
22/tcp   open          ssh
80/tcp   open          http
427/tcp  open          svrloc
443/tcp  open          https
902/tcp  open          iss-realsecure
5989/tcp open          unknown
8100/tcp open          unknown
123/udp  open          ntp
427/udp  open|filtered svrloc
MAC Address: 00:50:56:71:F9:14 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.45 seconds
```

4.7.3.3 VM_Admin accessible via l'interface NIC0 (réseau applicatif)
PC de test (nmap) relié à internet

```
> nmap -sUS 129.194.184.201
```

```
Starting Nmap 5.21 ( http://nmap.org ) at 2011-09-22 13:36 CEST
Nmap scan report for 129.194.184.201
Host is up (0.0012s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:0C:29:06:FE:65 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 17.95 seconds
```

4.7.3.4 VM_Worker accessible via l'interface NIC0 (réseau applicatif)
PC de test (nmap) relié à internet
> nmap -sUS 129.194.184.203

```
Starting Nmap 5.21 ( http://nmap.org ) at 2011-09-22 13:42 CEST
Nmap scan report for 129.194.184.203
Host is up (0.00094s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:0C:29:3C:C0:64 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 17.89 seconds
```

4.7.4 Savez-vous ce qui circule sur vos réseaux ? Volume ? Trafic légitime ? …
Voir l'annexe 4.

4.8 Applicatif

4.8.1 L'authentification : chaque connexion SSH vérifie l'identité du serveur (par sa clé d'hôte ~/.ssh/known_hosts) puis celle du client (par mot de passe ou clé publique ~/.ssh/authorized_keys).

Le tunneling SSH permet de sécuriser un service dont les informations circulent habituellement en clair (vPOPC)

4.8.2.1 Fichiers de configuration SSH

Tous les fichiers de configuration de OpenSSH se trouvent généralement dans `/etc/ssh/`
Pour ESXi, le serveur SSH est Dropbear et son dossier de configuration est `/etc/dropbear/`

`ssh_config`	Fichier de configuration du client SSH
`sshd_config`	Fichier de configuration du serveur SSH
`ssh_host_xxx_key`	Clé privé
`ssh_host_xxx_key.pub`	Clé publique

Sur le client `$HOME/.ssh/`

`known_hosts`	Contient les clés publiques des serveurs auxquels on s'est connecté.

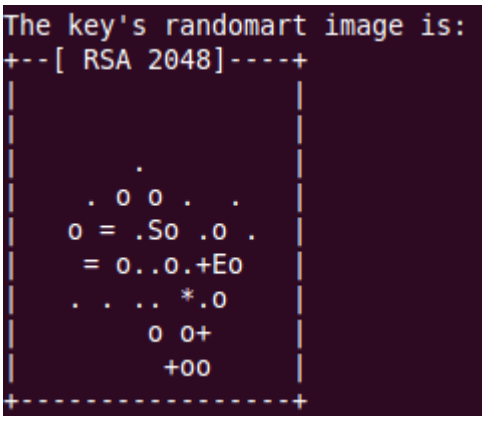
Sur le serveur `$HOME/.ssh/`

`authorized_keys`	Contient les clés publiques des clients distants autorisés.

4.8.2.2 Génération d'une paire de clés

Si aucune option particulière n'est spécifiée, les clés utilisées sont de type RSA 2048 bits.

```
> ssh-keygen
```



Options possibles : algorithme - longueur
-t (valeurs possibles : dsa ou rsa)
-b (nombre de bits, 512 étant le minimum)

4.8.2.3 Clé de session

Avec la version 2 du protocole SSH, le client génère une clé de session qui a une longueur d'au moins 128 bits.

4.8.2.4 Bonnes pratiques de configuration d'un serveur (`sshd_config`)

Port xxxxx
Utiliser un autre port que le 22 : Limite le nombre des attaques classiques.

AllowUsers ou **AllowGroups**
N'autoriser que les utilisateurs ou groupes définis à l'avance.

PermitRootLogin no
Ne pas autoriser l'utilisation du compte root avec ssh.

PermitEmptyPasswords no
Ne pas autoriser les mots de passe vides.

Protocole 2
N'autoriser que la version 2 du protocole ssh.

PasswordAuthentication no
En désactivant cette option on peut n'autoriser que les connexions avec les clés publiques.

MaxAuthTries X
Déconnecte un utilisateur après X tentatives de connexion. Utile pour se protéger contre les attaques de type « brute force »

4.8.2.5 Test de pénétration depuis internet

Pour auditer correctement un serveur que ce soit SSH ou autre, il faut se mettre dans la peau d'un pirate désireux de nuire et pouvoir reproduire sa façon de penser.

Une attaque suit presque toujours le même cheminement et je vais donc l'appliquer sur les serveurs du projet ViSaG :

1. Obtenir des informations publiques concernant la cible
2. Récupérer un maximum d'informations sur les ordinateurs ou réseaux cibles (OS, Ports ouverts, Versions, Applications…)
3. S'informer des différentes failles et « exploits » utilisables selon ce qu'on a trouvé au point 2.
4. Attaquer la cible.
5. Effacer toutes traces de notre passage.

Ce qui nous intéresse plus particulièrement dans cette démarche, ce sont les points 2 et 3 puisqu'ils définissent si un système est vulnérable ou pas.

1. Informations publiques

Nous connaissons les IP des machines à auditer :

129.194.184.96
129.194.184.201

2. Récupérer des informations sur les systèmes

Grâce au nmap de la partie 4.7.3, nous savons que 129.194.184.96 est inaccessible depuis internet et que 129.194.184.201 possède une ouverture uniquement sur le port 22 (Serveur SSH) et qu'il s'agit d'une machine virtuelle VMware vu son adresse MAC.

Essayons de récupérer des informations sur l'OS de cette machine.

```
> nmap -O -osscan-guess 129.194.184.201
```

```
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: printer|game console|switch|print server|specialized|PBX|broadband router
Running (JUST GUESSING) : Brother embedded (96%), Microsoft embedded (96%), 3Com embedded (96%), Sony
Vodavi embedded (94%), Nintendo embedded (93%), HP embedded (92%), Motorola VxWorks 5.X (92%)
Aggressive OS guesses: Brother MFC-7820N multifunction printer (96%), Microsoft Xbox game console (mod
boxMediaCenter) (96%), 3Com SuperStack 3 Switch 3870 (96%), Brother NC-130h print server (96%), Brothe
er (96%), Brother HL-5070N printer (96%), Sony FWD-40LX2F display card (96%), Vodavi XTS-IP PBX (94%),
me console (93%), HP ProCurve 2524 switch or 9100c Digital Sender printer (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.49 seconds
```

L'OS ne peut pas être défini

Comme le seul port ouvert est le 22, essayons d'obtenir des informations par ce chemin.
La commande ssh en mode verbose peut nous révéler quelques informations utiles.

```
> ssh -v 129.194.184.201
```

```
OpenSSH_5.5p1 Debian-4ubuntu6, OpenSSL 0.9.8o 01 Jun 2010
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Applying options for *
debug1: Connecting to 129.194.184.201 [129.194.184.201] port 22.
debug1: Connection established.
debug1: identity file /home/cedric/.ssh/id_rsa type -1
debug1: identity file /home/cedric/.ssh/id_rsa-cert type -1
debug1: identity file /home/cedric/.ssh/id_dsa type -1
debug1: identity file /home/cedric/.ssh/id_dsa-cert type -1
debug1: Remote protocol version 2.0, remote software version OpenSSH_5.5p1 Debian-4ubuntu4
debug1: match: OpenSSH_5.5p1 Debian-4ubuntu4 pat OpenSSH*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_5.5p1 Debian-4ubuntu6
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: server->client aes128-ctr hmac-md5 none
debug1: kex: client->server aes128-ctr hmac-md5 none
debug1: SSH2_MSG_KEX_DH_GEX_REQUEST(1024<1024<8192) sent
debug1: expecting SSH2_MSG_KEX_DH_GEX_GROUP
debug1: SSH2_MSG_KEX_DH_GEX_INIT sent
debug1: expecting SSH2_MSG_KEX_DH_GEX_REPLY
debug1: Host '129.194.184.201' is known and matches the RSA host key.
debug1: Found key in /home/cedric/.ssh/known_hosts:13
debug1: ssh_rsa_verify: signature correct
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: SSH2_MSG_NEWKEYS received
debug1: Roaming not allowed by server
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Trying private key: /home/cedric/.ssh/id_rsa
debug1: Trying private key: /home/cedric/.ssh/id_dsa
debug1: Next authentication method: password
visag@129.194.184.201's password:
```

Il s'agit donc d'un serveur OpenSSH en version 5.5 s'exécutant sur un Ubuntu. A partir de là, il est possible de faire l'installation du même OS pour trouver par exemple quels sont les comptes utilisateurs par défaut du système (voir 4.5.3). Ce qui va nous permettre de tenter des connexions avec ces comptes (brute force).
Un autre outil que j'ai utilisé pour cette partie, est Wireshark. Il nous permet d'obtenir les algorithmes utilisés par SSH grâce à l'analyse des paquets.

Capture sur la communication ssh avec 129.194.184.201 (VM_Admin)

Algorithme de chiffrement : aes128-ctr
Algorithme de hashage : hmac-md5 Algorithme jugé faible → utiliser SHA1, SHA2, …
Algorithme de compression : aucun

```
+ Frame 23 (82 bytes on wire, 82 bytes captured)
+ Ethernet II, Src: IntelCor_43:47:9f (00:1b:21:43:47:9f), Dst: Vmware_06:fe:65 (00:0c:29:06:fe:65)
+ Internet Protocol, Src: 129.194.184.82 (129.194.184.82), Dst: 129.194.184.201 (129.194.184.201)
+ Transmission Control Protocol, Src Port: 27066 (27066), Dst Port: ssh (22), Seq: 1056, Ack: 1696, Len: 16
- SSH Protocol
  - SSH Version 2 (encryption:aes128-ctr mac:hmac-md5 compression:none)
      Packet Length: 12
      Padding Length: 10
```

Par curiosité, j'ai également fait la capture pour l'hyperviseur même si celui-ci n'est pas accessible depuis l'extérieur.

Capture sur la communication ssh avec 10.1.250.100 (ESXi)

Algorithme de chiffrement : aes128-ctr
Algorithme de hashage : hmac-sha1-96
Algorithme de compression : zlib

```
+ Frame 427 (82 bytes on wire, 82 bytes captured)
+ Ethernet II, Src: Vmware_4c:a0:90 (00:0c:29:4c:a0:90), Dst: Vmware_71:f9:14 (00:50:56:71:f9:14)
+ Internet Protocol, Src: 10.1.40.33 (10.1.40.33), Dst: 10.1.250.100 (10.1.250.100)
+ Transmission Control Protocol, Src Port: 55870 (55870), Dst Port: ssh (22), Seq: 1032, Ack: 936, Len: 16
- SSH Protocol
  - SSH Version 2 (encryption:aes128-ctr mac:hmac-sha1-96 compression:zlib)
      Packet Length: 12
      Padding Length: 10
```

Pour terminer, j'ai utilisé l'outil par excellence de l'audit sécurité. Nessus permet grâce à des plugins configurables de tester toutes les faiblesses potentielles connues comme :

- les services vulnérables à des attaques permettant la prise de contrôle de la machine
- les fautes de configuration
- les patchs de sécurité non appliqués
- les mots de passe par défaut
- les services jugés faibles
- les dénis de service contre la pile TCP/IP

Résumé de l'analyse :

129.194.184.201

Scan Time

Start time :	Thu Oct 06 16:11:01 2011
End time :	Thu Oct 06 16:19:09 2011

Number of vulnerabilities

Open ports :	1
High :	0
Medium :	0
Low :	11

Remote host information

Operating System :	Linux Kernel 2.6 on Ubuntu 10.10 (maverick)
NetBIOS name :	
DNS name :	

Il détecte également un seul port ouvert mais il peut nous donner en plus l'OS de la cible. Et le plus important, aucune vulnérabilité grave n'a été trouvée.

3. S'informer des différentes attaques possibles et tester.

Pour commencer, on peut se rendre sur le site officiel d'openSSH et plus précisément sur la page relative à la sécurité[40]. Elle répertorie toutes les vulnérabilités connues selon les versions. Si notre version est présente, on peut commencer par attaquer grâce à ces informations.

A part ces failles, il existe de très nombreuses attaques possibles sur un système mais celles qui reviennent le plus souvent pour un serveur SSH sont :

- Attaque par Déni de Service
- Attaque par brute force
- Attaque par « downgrade » de version

Elles peuvent facilement être empêchées mais encore faut-il être au courant qu'elles existent. Il suffit en général de configuré le serveur SSH en respectant les bonnes pratiques de sécurité dont quelques points sont présents en §4.8.2.2.

Avec le client ssh disponible sous linux, nous pouvons faire quelques tests du serveur :

- Est-ce que la version 2 de SSH est la seule acceptée par l'hyperviseur et les VMs ?
  Connexion en utilisant l'option -1 (force l'utilisation de la version 1) nous retourne un message d'erreur.

```
> ssh -1 root@10.1.250.100 (ESXi)
> ssh -1 visag@129.194.184.201 (VM_Admin)
> ssh -1 visag@129.194.184.203 (VM_Worker)
```

```
cedric@ubuntu:~$ ssh -1 root@10.1.250.100
Protocol major versions differ: 1 vs. 2
cedric@ubuntu:~$ ssh -1 visag@129.194.184.201
Protocol major versions differ: 1 vs. 2
cedric@ubuntu:~$ ssh -1 visag@129.194.184.203
Protocol major versions differ: 1 vs. 2
```

- Est-ce que le serveur accepte des connexions non chiffrées ?
  Connexion avec l'option –c none (pour aucun chiffrement)

```
> ssh -1 visag@129.194.184.201 (VM_Admin)
```

```
cedric@ubuntu:~$ ssh -c none 129.194.184.201
No valid ciphers for protocol version 2 given, using defaults.
```

4.8.3 Ubuntu Server 10.10 (kernel 2.6.35)

[40] http://www.openssh.com/security.html

4.9 Stockage & Sauvegarde

4.9.1 Définir les biens à protéger, les risques, l'impact financier et les SLAs
Fonction du contexte applicatif

4.9.2 La sauvegarde de la configuration de l'hyperviseur peut être faite grâce à vSphere Management Assistant (vMA) qui est une VM fournie par VMware contenant des scripts pearl d'administration.

Télécharger la VM depuis le site VMware[41].

Déployer la VM sur l'hyperviseur dont on souhaite sauvegarder la configuration.

Entrer la commande suivante.

```
> vicfg-cfgbackup -s -server IP_Hyperviseur -username root -password PASS
nom_fichier_sauvegarde
```

Le fichier produit est une sauvegarde complète de la configuration.

Pour une restauration, même commande avec l'option –l

```
> vicfg-cfgbackup -l -server IP_Hyperviseur -username root -password PASS
nom_fichier_sauvegarde
```

4.9.3 La sauvegarde régulière des VMs n'est pas utile. Il suffit d'avoir les deux templates (VM_Admin.ovf & VM_Worker).

4.9.4 Tester la capacité à restaurer un système ou une VM dans le temps convenu selon le SLA
Procédure en cas de crash

Utiliser des OVF pour une installation simplifiée car ils ont été testé à l'avance et offrent un Admin et un Worker à 100% fonctionnels.

En cas de crash d'un nœud, la marche à suivre suivante peut être appliquée :

1. Réinstaller l'hyperviseur ESXi grâce à son CD ou PXE (~10min)

2. Remettre la configuration avec la méthode 4.9.2 (~5min)

3. Déployer les OVF de base (VM_Admin & VM_Worker) → voir Annexe 1 (~10min)

4.10 Supervision

4.10.1 Assurer que le niveau de sécurité de la plateforme reste le même dans la durée !!!

4.10.2 Comprendre les enjeux et les méthodologies

4.10.3 Eviter les comportements naïfs car l'adversaire est souvent malin

4.10.4 Disposer de références → profils systèmes et réseau

4.10.5 Surveiller l'occupation des ressources physiques

4.10.6 Surveiller les performances (qualité de service) afin de les comparer au SLA

4.10.7 Audit régulière (fichiers de config, …) pour détecter des anomalies comme rogue VM = VM non autorisée

4.10.8 Analyser (comprendre) les logs, contrôler la taille des logs, effectuer des sauvegardes

[41] http://communities.vmware.com/community/vmtn/server/vsphere/automationtools/vima

4.11 Recommandations selon VMware

VMware met à disposition un document disponible sur leur site officiel[42] listant quelques paramétrages et règles de sécurité à respecter.

Parmi elles, on peut retrouver notamment :

- *La compression répétée d'un disque virtuel peut le rendre inutilisable.*

  Paramétrage : isolation.tools.diskWiper.disable=TRUE
  isolation.tools.diskShrink.disable=TRUE

- *Limiter les accès distant à la console.*

  Paramétrage : RemoteDisplay.maxConnections=1

- *Désactiver la communication entre VMs par VMCI*

  Paramétrage : vmci0.unrestricted=FALSE

- *Limiter la taille et le nombre des fichiers de logs d'une VM*

  Paramétrage : log.rotateSize=1000000
  log.keepOld=10

- *Désactiver l'envoi d'informations de performances de l'hôte aux VMs*

  Paramétrage : tools.guestlib.enableHostInfo=FALSE

- *Eviter d'utiliser les certificats auto-signés d'ESXi et les remplacer par ceux de l'entreprise*

- *Activer la sychronisation NTP*

  Etc…

4.12 Références utiles

vSphere 4.1 Hardening Guide – Avril 2011
http://communities.vmware.com/docs/DOC-15413

End-to-End Virtual Environment Security – Edward L. Haletky – March 2010
http://communities.vmware.com/

Virtual Machine Security Guidelines - Version 1.0 - September 2007
http://benchmarks.cisecurity.org/tools2/vm/CIS_VM_Benchmark_v1.0.pdf

VMware ESX Server 3.x Benchmark Version 1.0 October 2007
http://benchmarks.cisecurity.org/tools2/vm/CIS_VMware_ESX_Server_Benchmark_v1.0.pdf

Autres documents
http://www.tdeig.ch/vmware/liens.pdf

[42] http://communities.vmware.com/servlet/JiveServlet/previewBody/15413-102-3-18829/vSphere%204.1%20Hardening%20Guide%20April%202011.pdf

Partie 5 : Etat de l'art

5.0 Objectifs

Cette partie résume (synthétise ?) les études préliminaires effectuées par l'auteur pour identifier les principaux risques.

Les principes d'ordonnancement sont basés sur la dépendance des publications et sur la capacité limitée de l'auteur à tout comprendre !

Les éléments mis en rouge seront repris dans mon analyse

5.1 Virtualisation et ou Sécurité – Nicolas RUFF (slides 11-20) – mars 2009

http://www.ossir.org/jssi/jssi2009/3A.pdf

Principaux risques :

- Humains
- Evasion (VM → VM, VM → hypervisor) – slide 13
- DoS – slide 14
- Hyperviseur – slide 17
- CPU – slide 20

Conclusion – slide 22

- Une architecture n'est jamais plus sûre après avoir été virtualisée
- Une architecture virtualisée ne se gère pas comme une architecture physique

GL

- Le risque humain ne doit pas être ignoré puisque qu'il accompagne tout processus d'entreprise, tout développement logiciel et toute infrastructure informatique.
  → L'être humain risque d'être pour longtemps encore le maillon faible dans la chaîne sécuritaire !
- Concernant l'affirmation "*Une architecture n'est jamais plus sûre après avoir été virtualisée*", le projet Terra (§4) insiste sur
  "The greater isolation of a VM, compared to a process in an ordinary OS, *can improve assurance on its own"*
  Il convient donc, dans le cas d'un serveur physique offrant divers services (DNS, web, …) de le scinder en VM spécifique pour lui garantir une meilleure sécurité intrinsèque.

5.2 Sicherheit in virtuellen Umgebungen mit VMware ESX – Roger Klose – juin 2008

http://www.ernw.de/content/e7/e181/e1212/download1218/DECUS_VirtSec_und_ESX_ger.pdf

- Problématique – slide 8
- Sicheren Virtualisierung – slide 10
- Avis – slides 12-13
- Evasion (VM → VM, VM → hypervisor, VM → Mgt) – slides 25-27
- Rogue VMs – slide 28
- Gartner "By 2010, unmanaged VMs will be as significant an issue to enterprises as unmanaged devices are in 2007 (0.9 probability) – slide 29
- Konfigurationsfehler – slide 30
- Compliance & Sicherheitsstrategie – slides 31-32
- Failles & exploits – slides 35-42
- Risques réseau (promiscuous, ARP, MiM) – slides 46-47
- Evasion VM → hypervisor – slides 50-51
  Voir http://taviso.decsystem.org/virtsec.pdf
- Katechismus – slide 64

GL :

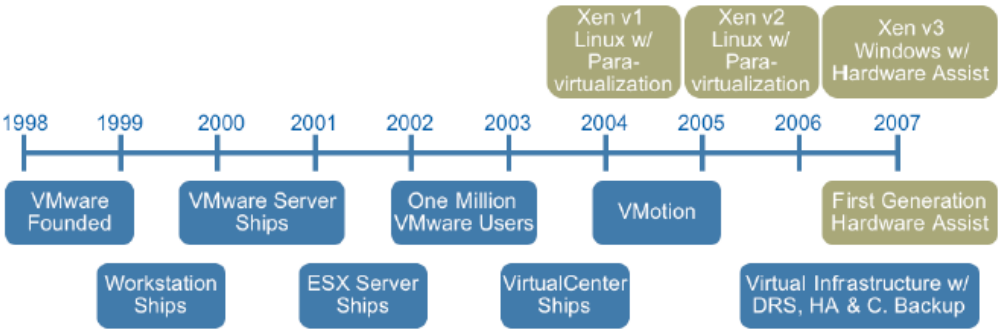
- Mon expérience, de 3 ans dans la virtualisation, confirme le danger d'héberger des VMs grises : elles ont été créées et démarrées dans l'urgence ; business oblige, mais elles ont le statut en test pour certains et en production pour d'autres)
- **Le risque de mauvaise configuration est très élevé** :
  - démarrer une VM avec memory size trop grand (il est 2 heures du matin et le brave ingénieur a entré un zéro de trop !!!)
  - placer une VM du mauvais côté du firewall
  - …

5.3 Terra: A Virtual Machine-Based Platform for Trusted Computing – (Garfinkel-Pfaff-Chow-Rosenblum Boneh)Standford – oct 2003

Extrait de http://suif.stanford.edu/papers/sosp03-terra.pdf :

- … commodity OS are complex programs (millions of lines of code) → offer low assurance
- … commodity OS poorly isolate applications from one another → the compromise of almost any application on a platform often compromises the entire platform
- … current platforms provide only weak mechanisms for applications to authenticate themselves to their peers.
- … current platforms provide no way to establish a trusted path between users and applications → an application for trading on financial markets has no way of establishing if its inputs are coming from a human user or a malicious program → human users have no way of establishing whether they are interacting with a trusted financial application or with a malicious program impersonating that application
- Terra supports today's OS & applications. Terra realizes this union with a trusted VM monitor (TVMM) = a high-assurance VM monitor that partitions a single tamper-resistant, general-purpose platform into multiple isolated VM
- The TVMM protects the privacy and integrity of a closed-box VM's contents
- Applications running inside a closed-box VM can tailor their software stacks to their security requirements.
- TVMM allows applications to cryptographically authenticate the running software stack to remote parties in a process called attestation
- At a high level, TVMM exports two VM abstractions : Openbox VMs provide the semantics of today's open platforms (run commodity OS and provide the appearance of today's general-purpose platforms), Closed-box VMs implement the semantics of a closed-box platform = Their content cannot be inspected or manipulated by the platform owner
- Terra's architecture is based on a virtual machine monitor (defined by Goldberg 1974)
- **The greater isolation of a VM, compared to a process in an ordinary OS,** can improve assurance on its own
- A VMM is a relatively simple program (Disco has only 13,000 lines of code [Bugnion-Devine-Rosenblum]), with a narrow, stable, well-defined interface to the software running above it.
- Unlike traditional OS, that must support filesystems, network stacks, etc., a VMM only needs to present relatively simple abstractions, such as a virtual CPU and memory
- As a result of these properties, VMMs have been heavily studied as an architecture for building secure OS [VAX VMM security kernel, KVM/370]. VMMs have long been a mainstay of mainframe computing [IBM], where their security has been leveraged for implementing systems for banking and finance, health
- The isolation properties of real-world VMMs such as that of the IBM zSeries have received intense scrutiny and been certified as conforming to the highest standards for assurance according to Common Criteria requirements
- **Attestation** allows an application running in a closed box to cryptographically identify itself to a remote party, that is, to tell the remote party what is running inside the closed box. This allows that party to put trust in the application, i.e. to have faith that the application will behave as desired
- A trusted path from the user to the application is essential for building secure applications
- The most important property Terra provides for improving application security is allowing applications to determine their own level of assurance
- … the fine-grained access controls of SELinux could be used to compartmentalize a more sophisticated application with many components …
- We built a prototype of the TVMM using VMware GSX Server 2.0.1 with Debian GNU/Linux as the host OS. Neither Debian nor VMware GSX Server is suitably high assurance for a real TVMM, but they form a convenient platform for experimentation.

Pour mémoire



GL :

- Dans quelle mesure Terra a influencé les versions ESX ???

5.4 Server Virtualization Security: 90% Process,10% Technology – Galen Schreck – juillet 2008

http://i.i.com.com/cnwk.1d/html/itp/Tripwire-Forrester_VRT.pdf

- Perhaps the biggest concern with securing x86-based virtual server environments is updating existing management and security processes to cope with relatively new technology. There are no known exploits that can critically compromise a virtual server based on Citrix/Xen, Microsoft, or VMware technologies — although the vendor community is in the process of improving control and visibility of the underlying virtualization layer. In the interim, Forrester believes that updated security practices will adequately protect most organizations' systems.
- Like any new technology, poor implementation and process may create some vulnerability. In the case of server virtualization, the risk would not be loss of security between VMs but rather access to the console environment, poor patch compliance, poor network architecture, and overall quality of service.
- Virtual network : the same design principles that apply to physical networks apply to virtualized networks within VMware ESX
- The virtualization kernel : security of the virtualization kernel or hypervisor itself is paramount but there are really no user-serviceable parts inside.
- Many people confuse the kernel itself, which contains a stripped-down operating system, with the console operating system, which runs as a guest on top of the kernel, and is your primary interface to the hypervisor.
- Just how much of the original operating system remains in the kernel depends on the vendor — VMware's being the smallest …. The security of the kernel is critical, and it's generally accepted that the less code that runs here the better, since it provides lower potential for vulnerabilities.
- VMware ESXi is the smallest kernel yet, and at 32 MB it only communicates with the outside world via an API, as there is no Linux-based console at all.
- **The biggest challenge with the virtualization kernel's security is the difficulty in knowing if it has been compromised.**
- While there are no existing vulnerabilities, there is always the chance that one will be eventually discovered. To counter this risk, VMware has created an API — called VMsafe — that allows security tools to inspect the underlying hypervisor.
- Forrester believes that competitors like Microsoft and Citrix will have to quickly follow suit, since security tools that run inside of guest operating systems can't see the underlying hypervisor.
- In other words, without access to the kernel, software running in a guest operating system can't guarantee that the underlying system has not been compromised and that isolation between the guests is still intact.

GL :

- Tout système (OS, hyperviseur, …, ABC) présente le risque d'un défaut conceptuel dû à un mauvais *design* ou une mauvaise spécification :
  - Certain serveur (web, …, SQL) ne teste pas (ou mal) les données entrantes
  - Certain serveur ne gère pas l'authentification du client de manière permanente
  - …
  - L'excellent site https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project illustre la réalité des serveurs web en matière de risques :
    1. Injection
    2. Xxx
    3. Broken Authentication and Session Management

- Le cauchemar de tout spécialiste en sécurité se résume à la notion de **faux négatifs** *(false negative)* qui correspondent à des intrusions non détectées.
- Le meilleur dispositif de détection d'intrusion ABC n'est pas à l'abri car il risque de fonctionner comme un vulgaire anti-virus selon le modèle *black-list* en n'étant capable d'identifier que ce qui est connu !!!

- Pour avoir développé du logiciel de communication RNIS, je puis affirmer que les phases de spécifications et de tests sont vitales.
- Ces produits ont donc bénéficié durant des décennies d'une confiance totale *(trust)* de leurs utilisateurs comme vous le faites aujourd'hui avec les périphériques USB

- Dans ces 2 contextes (RNIS, USB), les normes sont ouvertes et les protocoles ont été spécifiées et testés par des experts (formalisme, exempt d'interblocage, de fuzzing, …)

5.5 Virtualization and Security Boundaries – Mike Lococo – juillet 2009

http://mikelococo.com/files/2009/virtualization-and-security-boundaries.pdf

- This guide presents a taxonomy of virtualization technologies that is helpful in security analysis, an overview of attacks that are possible in virtualized environments, a summary of current best practices, and a list of unresolved challenges
- This guide is the result of a survey of existing public literature
- Software-based partitioning strategies are employed by many commodity server and … ESX & KVM
- Jailbreak Attacks, aka Escapes (VM Escape dans ce document) = any action which results in a user or administrator of one guest gaining unauthorized access to the underlying host …may be characterized as a jailbreak attack
- At the time of this writing, there are no publicly known examples of jailbreak attacks being used in the wild against any major virtualization platform.
- Migration Attacks : many virtualization technologies provide the capability to move a VM from one physical host to another physical host. Typical attacks require the attacker to have access to the network where migrations are occurring
- Classify Data and Systems : the primary security function of a virtualization platform is to enforce security boundaries between different guests, and between guests and the host.
  - Classify data according to the risk of a breach in the confidentiality, integrity, accessibility, or auditability of that data.
  - Classify virtualized guest operating systems according to the data that they store and process
  - Classify hosts according to the guests that run on them
  - Classify networks and storage appropriately
- Evaluate Virtualization Products Before Deployment = perform a risk assessment on new virtualization technologies before deploying them. … determine what boundaries they can be trusted to enforce. Periodically review these risk assessments … evolving product and threat
- Many excellent hardening guides exist
- Containing a Compromised Host : if used to consolidate systems of different classifications, virtualization platforms have the potential to aggregate an unpre edented amount of risk into a single system component. It is desirable, therefore, for that component to employ a layered security strategy that allows faults to be contained to a single host or set of hosts, even if those hosts are compromised. Unfortunately, most vendors are currently focused on hardening the perimeter of their platforms and do not consider the containment of a malicious host to be a viable threat scenario. VMWare's work on the VMSafe API is an early step toward improving detection of compromised guests, but at this time no vendor using a software-based partitioning approach appears to have engaged the issue of containing a malicious host, or an attacker with access to the migration and storage networks.
- The VMSafe API is said to allow trusted inspection of network, memory, disk, and processor activity of guests and will enable a variety of monitoring appliances to be

GL :

- Concernant *" there are no publicly known examples of jailbreak attacks*" voir mon analyse du §4 : faux négatifs
- Dans "*Classify Data*", je comprends de placer les objets dans diverses catégories.
  La classification des données en Suisse est soumise à la LPD
  http://www.admin.ch/ch/f/rs/235_1/index.html car l'information de l'entreprise appartient aux diverses personnes qui la composent mais pas à l'ingénieur système qui gère son parc informatique.
- Dans "*Classify Data & Systems*", je remplacerais par Classify Assets pour faire le lien avec *Risk Management*
- L'évaluation de nouvelles solutions est un passage obligé ; ce labo peut et veut aider des entreprises dans l'utilisation de la solution KVM
- Le § 4.11 énumère divers *Hardening Guide*

5.6 Empirical Exploitation of Live Virtual Machine Migration

http://www.blackhat.com/presentations/bh-dc-08/Oberheide/Whitepaper/bh-dc-08-oberheide-WP.pdf

- Live migration of virtual machines (VMs), the process of transitioning a VM from one virtual machine monitor (VMM) to another without halting the guest operating system, often between distinct physical machines, has opened new opportunities in computing [redacted]
- [redacted] A virtual machine monitor that incorporates a vulnerable implementation of live migration functionality may expose both the guest and host operating system to attack and result in a compromise of integrity.
- This paper explores attacks against live virtual machine migration in the context of these three threats. We present several practical attacks against the migration functionality of the latest versions[1] of the Xen [1] and VMware [12] virtualization products and develop a tool to automate the manipulation of a guest virtual machine's memory during live migration. [redacted]
- While various virtual machine monitors have different wire protocols for live migration, the underlying algorithms are similar. Live migration techniques [5, 9, 17] usually begin by copying memory pages of the VM across the network from the source VMM to the destination while the VM continues to run within the source VMM. This process continues as pages are dirtied by the VM. When the source VMM reaches a threshold
- This paper has empirically demonstrated how two of the most popular and widely deployed VMMs, Xen and VMware are vulnerable to practical attacks targeting their live migration functionality. These threats are cause for concern and require that appropriate solutions be applied to each class of live migration threats.

  In order to support the secure migration of virtual machines, mutual authentication of the source and destination VMMs, as well as any management agents, must be performed to protect the control plane communications. Flexible access control policies should allow administrators to manage migration privileges. The data

GL :

Cette étude confirme les risques de fuite lors d'opérations vMotion (ESXi) ou Live Migration (KVM)

5.7 Vulnérabilités – GL – juin 2011

Ce paragraphe utilise l'excellent site www.secunia.com pour identifier les principales failles rendues publiques en 2011 et 2010 sur le produit ESXi

5.7.1 VMware … / ESXi e1000 Driver Packet Filter Bypass Security Issue
mount.vmhgfs in the VMware Host Guest File System (HGFS) in VMware … ESXi 3.5 though 4.1 allows **guest OS users** to determine the existence of **host OS files and directories** via unspecified vectors.
http://secunia.com/advisories/cve_reference/CVE-2011-2146/

5.7.2 VMware … / ESXi OpenSSL Vulnerabilities
Double free vulnerability in the ssl3_get_key_exchange function in the OpenSSL client (ssl/s3_clnt.c) in OpenSSL 1.0.0a, 0.9.8, 0.9.7, and possibly other versions, when using ECDH, allows context-dependent attackers to cause a denial of service (crash) and possibly execute arbitrary code via a crafted private key with an invalid prime
http://secunia.com/advisories/cve_reference/CVE-2010-2939/

5.7.3 VMware ESXi curl Security Issue
VMware has acknowledged a security issue in VMware ESXi, which can be exploited by malicious people to cause a DoS (Denial of Service) or compromise a vulnerable system
http://secunia.com/advisories/43313/

5.7.4 VMware … / ESXi Service Location Protocol Daemon Denial of Service
The extension parser in slp_v2message.c in OpenSLP 1.2.1, and other versions before SVN revision 1647, as used in Service Location Protocol daemon (SLPD) in VMware … ESXi 4.0 and 4.1, allows remote attackers to cause a denial of service (infinite loop) via a packet with a "next extension offset" that references this extension or a previous extension. NOTE: some of these details are obtained from third party information.
http://secunia.com/advisories/cve_reference/CVE-2010-3609/

GL

- Une partie des failles sont connues (publiques)
- Confirmation du §4 : les risques semblent faibles
- VMware a tendance à désigner l'Open Source comme le responsable des problèmes → ne plus l'utiliser ou mieux le tester !

5.8 VMware VMsafe Partner Program

http://www.vmware.com/technical-resources/security/vmsafe/security_technology.html

- The VMsafe program is extended to trusted partners for integration with VMware vSphere environments, providing partners with visibility and control of virtual resources. Customers benefit from a choice of virtualization security solutions that best meet their needs.
- The VMsafe program enables partners to build security solutions based on VMware's first generation introspection technologies for network, CPU, memory and storage.

GL : Voir §2.4.4

5.9 Stealthy Deployment and Execution of In-Guest Kernel Agents - Symantec Research Labs - Date ???

http://www.blackhat.com/presentations/bh-usa-09/CONOVER/BHUSA09-Conover-SADEintoVM-PAPER.pdf

- This paper describes the design, implementation and evaluation of a stealthy agent deployment and execution mechanism called SADE
- Conceptually, SADE is designed to inject a piece of arbitrary binary code into a VM's kernel, and then execute it within the VM's kernel address space in a way that does not require installing any new code in that VM
- Our work takes advantage of the functionality of VMware's vmsafe-Mem/CPU API and does not involve any custom modifications to hypervisors, hardware, or guest operating systems.
- VMsafe consists of a set of components and SDKs that support third-party security add-ons for VMware's vSphere environment. Instead of a single technology, the original version of VMsafe consists of the following 5 components, each of which corresponds to a particular SDK
- To inject a piece of binary code C into the kernel address space of a user VM requires solving two problems: (1) allocation of a kernel memory region to hold C, and (2) resolving all the references in C to kernel service functions. SADE effectively addresses these two problems and runs on the virtual appliance VM.
- The proposed virtual appliance execution architecture is a promising way to remove redundant resource usage in future data centers, where a single physical server can host dozens or even hundreds of virtual machines.
- A key design decision in this virtual appliance architecture is how to enable a virtual appliance to perform VM-specific operations, such as accessing and modifying kernel service functions exported by a VM's kernel.
- An agentless architecture is more secure, less flexible and easier to deploy, whereas an agentful architecture is more flexible, less secure and more difficult to deploy.
- This paper describes the SADE system that simultaneously achieves the flexibility of the agentful architecture and the security and ease of deployment of the agentless architecture.
- The key building blocks of SADE are its ability to resolve calls to kernel service functions without relying on the kernel loader, and its code injection technique that ensures the injected code is invoked in a proper execution context.
- We have built a fully operational SADE prototype using the VMsafe API and applied it to a security-based virtual appliance application, which scans the kernel memory of the VMs against a string signature database.
- In summary, the research contributions of this work are
  - A stealthy code injection mechanism that allows one VM to inject binary code into another VM without involving the target VM or requiring pre-installation of any other code on the target VM.
  - A remote code execution mechanism that allows one VM to run a piece of code on another VM using a proper execution context without requiring any help from the target VM.
  - A fully working SADE prototype for a Linux-based source VM and aWindows XP-based target VM, and a demonstration of its usefulness through a complete virtual appliance application, a shared kernel memory scanner running on a virtual appliance VM that scans the kernel memory for other user VMs running on the same physical machine.

GL :

- Devant la complexité de SADE, le profane que je suis en programmation Kernel se rappelle cette magnifique affirmation :
  "Any problem in computer science can be solved with another layer of indirection. But that usually will create another problem"
  David Wheeler - Computer Scientist
- J'ai une autre raison de quitter ESXi pour KVM

5.10 Le point de vue d'Intel

http://software.intel.com/sites/oss/pdfs/VT_Security_Whitepaper.pdf

§2. Hardware, Software, and Policy Measures to Mitigate Security Challenges

- The maturation of virtualization security is marked by an extensive and growing synergy between **hardware and software**. Successive generations of Intel hardware platforms offer new virtualization and security features, and collaboration between Intel and the major providers of VMMs from a very early point in the development of those hardware features ensures their rapid support by the virtualization software ecosystem.
- As with most computing technologies, security challenges will always be part of the virtualization landscape, and early and ongoing collaboration and testing will result in a higher quality, increasingly robust set of solutions.

§2.1. Hardware Features that Enhance the Protection of Virtualization Software

- The growing set of hardware-based extensions to **Intel VT** offers enhanced protection capabilities to virtualization software, particularly by reducing the VMM size and complexity while increasing functionality and improving performance.
- Simplifying the VMM is a means to enhance security. By offloading many tasks from the VMM and handling them in **hardware**, Intel VT increases tamper resistance as well as reduces the attack surface associated with the software platform.
- Intel VT allows guest OSs to operate at their native privilege levels and avoids the need for the VMM to perform certain complex software operations. The resultant simpler VMM presents a reduced attack surface. For example, when an instruction from a guest OS calls a critical platform resource, Intel VT's ability to allow that OS to run at its native Ring-0 privilege avoids system faults or wrong responses that could potentially be exploited or otherwise affect operations.
- Ongoing enhancements in Intel VT that reduce the scope of tasks that the VMM must complete in software continue to dramatically reduce the size and complexity of VMMs, allowing hardening efforts to be easier and more successful.
- Combined advances in hardware and software features help to provide the basis for robust protection of the virtualized environment. Intel VT and security technology features that are particularly important from a protection perspective include the following:
  • Intel VT-x provides a new privilege space in which the VMM can operate, reducing the size and complexity of the VMM for a smaller trusted computing base and reducing the attack surface. Extended Page Tables is an example of a capability that eliminates the requirement for page tables to be maintained in the VMM and hence reduces the VMM footprint. Features such as Descriptor Table Exiting that are part of Intel VT-x when enabled by software help maintain the integrity of guest VMs by preventing malicious guest software from relocating key OS data structures, such as interruptvector tables

2.2. Comparing the Security Model of a VMM to That of a Traditional OS

- VMMs and traditional OSs are similar in the sense that both control system resources, but VMMs are typically simpler and have a more limited set of interfaces. Also, the impact of a security breach to a VMM is much greater than a breach to an OS, since a breach to a VMM could involve multiple OSs, and the need to secure the guest OS itself is still present
- As shown in Figure 3, VMMs designed for Intel VT operate in a dedicated processor mode called "VMX Root" that provides executive control over platform resources and guest OSs while allowing those guest OSs to run at their native privilege level.
- The security model of a traditional OS assumes that it has full and unique access to hardware resources, such as processor, memory, and I/O devices. Its security model with regard to those resources is concerned primarily with managing access among applications.
- A VMM's management of such access among guest OSs provides an added point of control over the system as a whole, and VMMs have the goal of providing a similar level of protection and **isolation** as that provided by stand-alone physical platforms for their system software.
- IT organizations should also note, however, that the overall complexity of a virtualized system (which includes a VMM and multiple guest OSs) is higher than a single-OS unvirtualized one. That added complexity could be a significant consideration in terms of the overall security of a virtualized environment.
- Both VMMs and traditional OSs must be sufficiently hardened to resist intrusion attempts and malware exploits. In addition they must be properly monitored and instrumented to detect any compromises that do occur.
- One such significant area of research concerns protecting guests in the virtualized environment with VM introspection, the practice of transparently inspecting VMs at a low level from the outside as they run, to verify that their contents have not been compromised by intruders.

GL

- L'accès aux sources de Linux KVM va être un travail intéressant pour valider cette sécurité (matérielle et logicielle)

5.11 VMM : la clé de voute de la sécurité

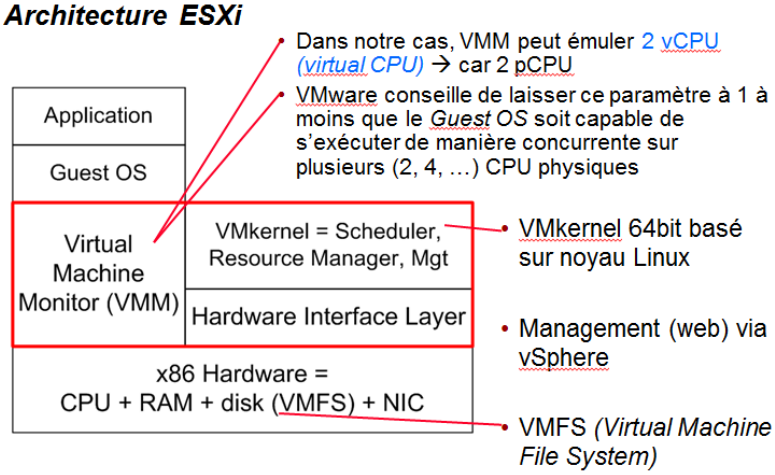
Le lecteur pourra compléter ses connaissances du monde x86 avec les excellents articles très pédagogiques développés par David Decotigny & Thomas Petazzoni → http://sos.enix.org/

Cette partie tente d'expliquer le rôle ainsi que le fonctionnement de VMM (Virtual Machine Monitor)

Le système d'exploitation (Guest OS) **croit** accéder au matériel (CPU, RAM, …) alors qu'il communique avec VMM

VMM émule la couche matériel ; on parle de vCPU = virtual CPU, …

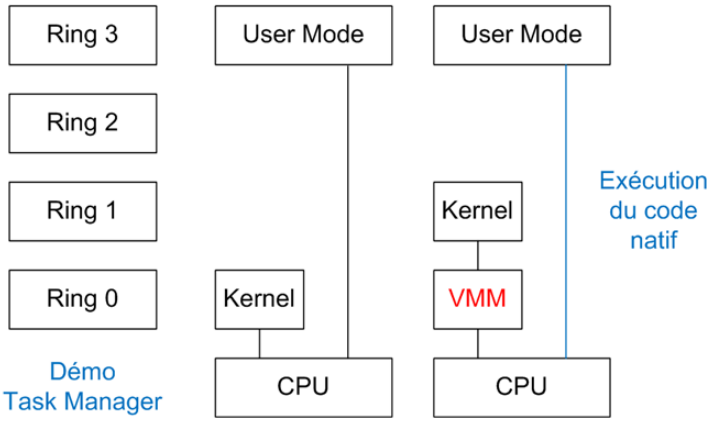
Architecture ESXi



La figure ci-contre représente verticalement les 4 niveaux (ring) d'exécution du CPU; le niveau 0 étant le plus privilégié (le CPU est autorisé à exécuter l'ensemble du jeu d'instruction)

Un système d'exploitation comme Windows ou Linux utilise 2 niveaux : le noyau (kernel) gère l'ordonnancement des processus et l'accès au matériel (CPU, RAM, …) alors que les processus utilisateur bénéficient d'un jeu d'instruction réduit (impossible par exemple d'accéder à la mémoire physique)

Exécution du code x86



Principe de fonctionnement

1. Le système d'exploitation (ainsi que le applications) ne subissent aucune modification
   Les fichiers binaires exécutés (OS + applications) sont les mêmes avec ou sans virtualisation
2. Le code à exécuter en mode Ring 3 accède directement au CPU physique car le jeu d'instruction est limité
3. VMM intercepte certaines instructions privilégiées comme par exemple l'écriture à une adresse mémoire

Binary Translation or hardware support for CPU virtualization

Extrait de http://www.vmware.com/files/pdf/perf-vsphere-monitor_modes.pdf

Included in VMware® vSphere™ 4.0 is VMware ESX™ 4
All versions of ESX comprise two components: the virtual machine monitor (VMM) and the kernel.
The monitor is a thin layer that provides virtual x86 hardware to the overlying operating system.
The virtual hardware is called a virtual machine and the operating system it runs is called the guest.
Prior to the introduction of hardware support for virtualization, the VMM could only use software techniques for virtualizing x86 processors and providing virtual hardware.
This software approach, binary translation (BT), was used for instruction set virtualization and shadow page tables for memory management unit virtualization
Today, both Intel and AMD provide hardware support for CPU virtualization with Intel VT-x and AMD-V, respectively.
More recently they added support for memory management unit (MMU) virtualization with Intel EPT and AMD RVI.
In the rest of this paper, the following are referred as follows: hardware support for CPU virtualization as hardware virtualization (HV), hardware support for MMU virtualization as hwMMU, and software memory management unit virtualization as swMMU.

Aujourd'hui VMM est essentiellement implémentée au niveau du matériel par Intel (ou AMD) ; certaine solution récente de virtualisation comme KVM exige ce support matériel.

VMX root operation mode, or “Ring -1”.

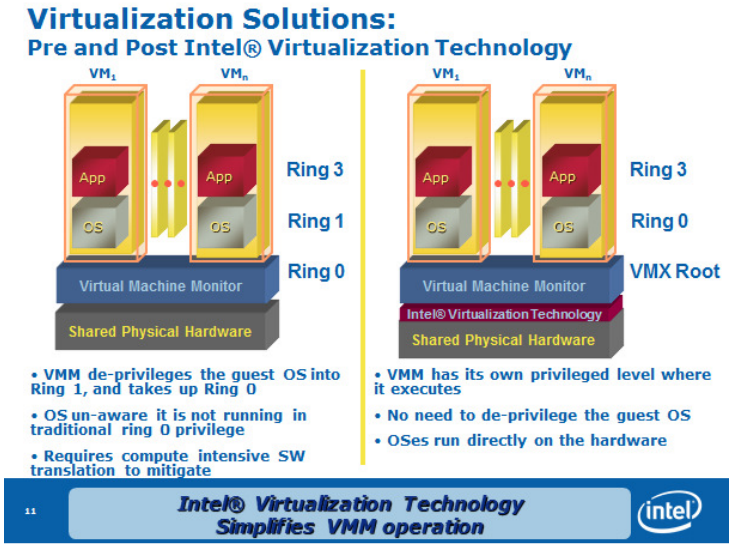
VMCS – the VM control structure.

10 Instructions.

Changes to the “normal” (VMX non-root) mode operation for some instructions.

VMM can specify what interrupts / exceptions it wants notification.

Shadows for control register reads.

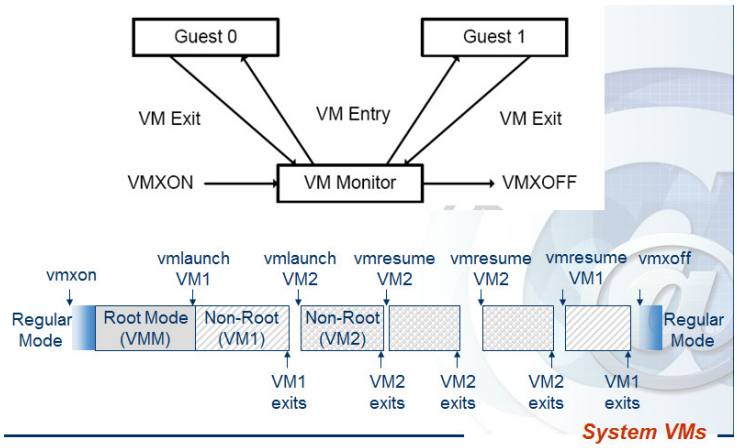


vmxon → activates VT mode (BIOS)

VM Entry → transition from VMM to Guest
- Enters VMX non-root operation
- Loads Guest state from VMCS
- vmlaunch instruction used on initial entry
- vmresume instruction used on subsequent

VM Exit → transition from Guest to VMM
- Enters VMX root operation
- Saves Guest state in VMCS
- Loads VMM state from VMCS

Vmxoff



Extrait du rapport de Pierre Kunzli 2010 sur l’étude du mode Intel VT

Les appliances destinées à ESXi contiennent des informations de configuration préférées pour la VM en question, ESXi prendra la première option compatible avec le matériel à disposition. Il est possible de forcer un mode d’exécution (dans la mesure des limitations techniques évoquées et du matériel à disposition) de la manière suivante : click droit sur VM -> edit settings -> options -> CPU/MMU Virtualization

Il est possible de savoir dans quel mode fonctionne une VM en consultant les logs en recherchant les lignes commençant par "MONITOR MODE" :

```
Mar 22 13:49:38.618: vmx| MONITOR MODE: allowed modes : BT HV
Mar 22 13:49:38.618: vmx| MONITOR MODE: user requested modes : BT HV HWMMU
Mar 22 13:49:38.618: vmx| MONITOR MODE: guestOS preferred modes : HWMMU HV
    BT
Mar 22 13:49:38.618: vmx| MONITOR MODE: filtered list : HV BT
Mar 22 13:49:38.618: vmx| HV Settings: virtual exec = 'hardware';
    virtual mmu = 'software'
```

On voit dans ce cas que le processeur permet soit d’utiliser Intel VT, soit la BT, que l’utilisateur souhaite utiliser soit BT soit Intel VT ainsi que le support hardware de la virtualisation de la MMU et que le guest OS souhaite utiliser le support hardware de la virtualisation de la MMU et Intel VT en préférence de la BT. ESX choisit alors d’utiliser Intel VT, mais utilise une MMU virtuelle software, le support hardware n’étant pas intégré dans le processeur utilisé (Core 2 duo).

Matériel disponible → http://www.tdeig.ch/vmware/Montage_PC_Gigabyte.pdf

Voir aussi la structure logicielle dans Linux KVM → http://www.tdeig.ch/kvm/pasche_R.pdf pages 44-45

Au niveau de la **gestion mémoire**, il convient de distinguer **3 types d'adresse : logique – linéaire – physique**
La figure ci-dessous ne prend pas en compte la virtualisation.

Adresse logique
Au niveau logiciel (code assembleur, code C++ compilé, …), le jeu d'instructions x86 utilise un **segment** et un déplacement (offset)

Adresse linéaire
Un processeur IA-32 est capable d'adresser 4 Gbyte

L'espace physique est découpée en **page**
L'unité de pagination traduit les adresses linéaires en adresses physiques

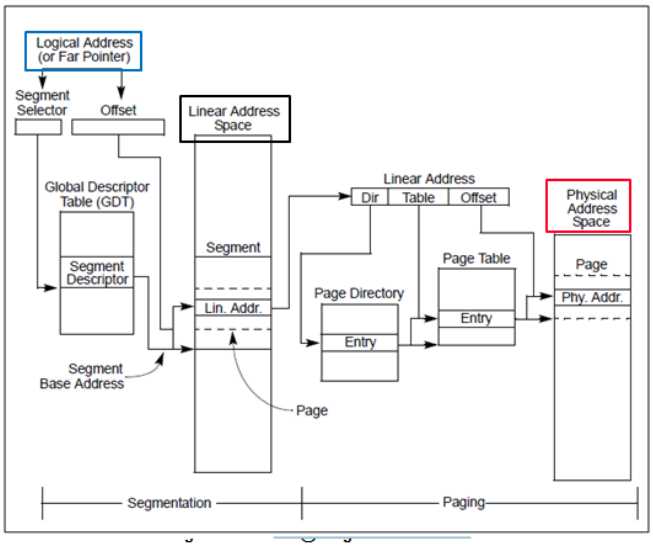
Adresse physique générée par les signaux électriques sur le composant mémoire RAM

Segmentation and Paging

Logical Address
= segment + offset

Linear Address Space = 4 GByte

Physical Address to RAM hardware



VIRCL 2010

Selon http://sos.enix.org/wiki-fr/upload/SOSDownload/sos-texte-art4.pdf

Le terme "pagination" vient de "page". En informatique, une "page" de mémoire (fictive, physique, ou sur le disque dur) est un bloc de taille donnée dépendante de l'architecture, de l'ordre de quelques kilo-octets le plus souvent

Pagination

- Les OS actuels (Linux, Windows) gèrent la pagination et sont libres de l'implémentation utilisée
- Dans une architecture virtualisée; seul l'hyperviseur (ESX, …) accède au matériel (pRAM) et peut gérer l'espace mémoire
- L'OS ne voit pas la couche de virtualisation
  → Il croit donc gérer l'espace mémoire
- Le noyau assure la transparence via un mécanisme appelé *Shadow Page*

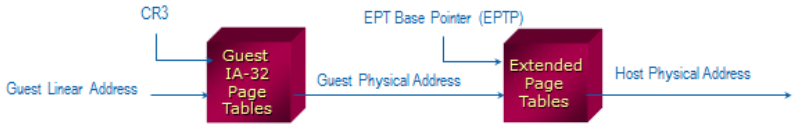
Problématique sans EPT
Voir slides 22-26
http://altair.snu.ac.kr/newhome/kr/course/system_software/2005/SystemVM.ppt

Extended Page Tables (EPT)

A VMM must protect host physical memory
- Multiple guest operating systems share the same host physical memory
- VMM typically implements protections through "page-table shadowing" in software

Page-table shadowing accounts for a large portion of virtualization overheads
- VM exits due to: #PF, INVLPG, MOV CR3

Extended Page-Table
- A new page-table structure, under the control of the VMM
  - Defines mapping between guest- and host-physical addresses
  - EPT base pointer (new VMCS field) points to the EPT page tables
  - EPT (optionally) activated on VM entry, deactivated on VM exit
- Guest has full control over its own IA-32 page tables
  - No VM exits due to guest page faults, INVLPG, or CR3 changes



30 *Extended Page Tables will reduce these overheads!* intel

5.12 Rootkit, Malicious Code, IDS

Guest-Transparent Prevention of Kernel Rootkits with VMM-based Memory Shadowing – Date ?
http://faculty.qu.edu.qa/rriley/pubs/RAID08.pdf
Kernel rootkits pose a significant threat to computer systems as they run at the highest privilege level and have unrestricted access to the resources of their victims.
In this paper we present a kernel rootkit prevention system called NICKLE which addresses a common, fundamental characteristic of most kernel rootkits: the need for executing their own kernel code.

Stealthy Malware Detection Through VMM-Based "Out-of-the-Box" Semantic View Reconstruction - 2007
http://www.csc.ncsu.edu/faculty/jiang/pubs/CCS07.pdf

Analysing Malicious Code: Dynamic Techniques – Lars Haukli – juin 2007
http://daim.idi.ntnu.no/masteroppgaver/IME/ITEM/2007/3660/masteroppgave.pdf
In this project, the study of methods and techniques to analyse malicious code will be performed.
How to combine techniques in order to detect different flavours of malware, as well as how to automate (parts of) the analysis process will be emphasized.
The primary focus will be on the analysis of binary code in the form of PE or PE+, but it is believed that other file formats will require more or less the same techniques and the project could of course be extended to include other file formats running on other platforms than Microsoft's.

A Virtual Machine Introspection Based Architecture for Intrusion Detection – Garfinkel & Rosenblum - Date?
http://suif.stanford.edu/papers/vmi-ndss03.pdf
In this paper we present an architecture that retains the visibility of a host-based IDS, but pulls the IDS outside of the host for greater attack resistance. We achieve this through the use of a virtual machine monitor.
Using this approach allows us to isolate the IDS from the monitored host but still retain excellent visibility into the host's state.
The VMM also offers us the unique ability to completely mediate interactions between the host software and the underlying hardware.

Asynchronous pseudo physical memory snapshot and forensics on paravirtualized VMM using split kernel module – Date ?
http://www2.nict.go.jp/old/y/y212/ruo/pdfs/ICISC07_RuoAndo.pdf
VMM (virtual machine monitor) based system provides the useful inspection and interposition of guest OS.
With proper modification of guest OS, we can obtain event-driven memory snapshot for malicious code forensics.
In this paper we propose an asynchronous memory snapshot and forensics using split kernel module.
Our split kernel module works for virtualized interruption handling, which notifies malicious fault, illegal system call and file access.
On frontend, we insert virtualized interruption into source code of MAC (mandatory access control) module, fault handler and gcc-extension.
Then, Backend kernel module receives the asynchronous incident notification.
In experiment, we take RAM snapshot of LKM-rootkit installation using system call extension.
Frequently appeared n-grams is extracted by weighted resolution in order to find memory blocks which is used by LKM-rootkit.
Proposed system can detect unknown malware (malicious software) of which name is not matched by signature.
Also, it is showed that we can find evidence of malware from memory snapshot by linear extraction algorithm.

Towards a VMM-based usage control framework for OS kernel integrity protection - 2007
http://portal.acm.org/citation.cfm?id=1266852

GL :

- Détecter la présence de rootkit peut sembler une défense bien tardive … naïve ?
- L'utilisateur d'un hyperviseur doit pouvoir garantir qu'il est exempt de rootkit
- La défense logique consiste donc à contrôler l'intégrité de cet hyperviseur
- L'opération est facile lors de l'installation car les fichiers exécutables sont généralement signés
- Existe-t-il un moyen efficace et sure de contrôler périodiquement l'intégrité de l'hyperviseur chargé en mémoire RAM ??? → A étudier dans un prochain travail (thèse) de Master

Annexe 1 : Guide d'installation & configuration d'un nœud

Chaque nœud repose sur un hyperviseur du type ESXi et comprend 1 + X machines virtuelles

- VM Admin
- VM_WorkerX

Contrôle d'intégrité lors de l'installation

Des OVF pour chaque VM sont fournis par Fribourg :
http://valentin.clement.home.hefr.ch/

Le contrôle d'intégrité peut se faire grâce à un hash md5 de chaque fichier. Cette valeur doit être fournie par Fribourg.

Remarque : Pour une installation à partir de zéro, utiliser la documentation faite par Valentin :
http://valentin.clement.home.hefr.ch/visag/user_manual_addon.pdf

Installation avec les OVF :

Pour l'admin :

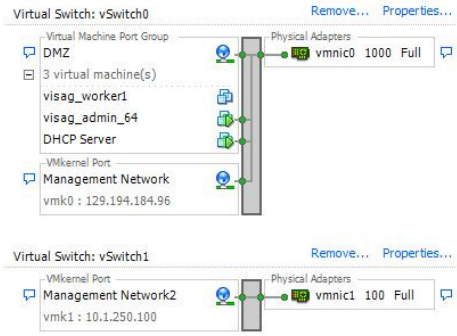
1. Déploiement de l'OVF
2. Démarrage de la VM
3. Modification de /etc/hosts
   IP ESXi nom_machine.localdomain
4. Reboot
5. Compilation de POP-C++ dans /home/visag/versioning/popcpp
   - make
6. Installation de POP-C++
   - make install
7. Démarrage du service POPC
   - SXXpopc start

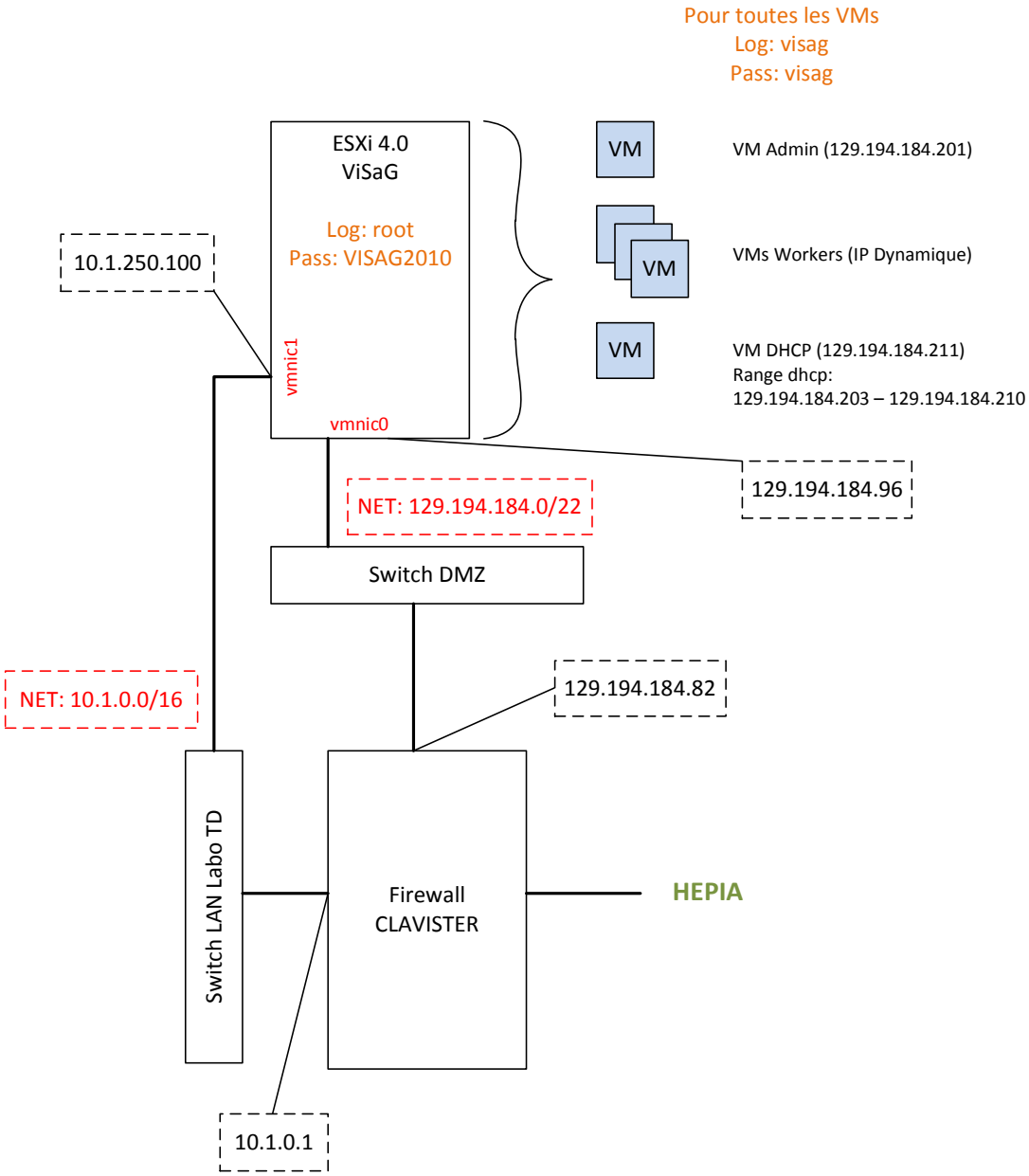
Pour le worker :

1. Déploiement de l'OVF
2. Démarrage de la VM
3. Forcer le renouvellement de l'IP + arrêt de la VM
   - sudo dhclient –r && sudo shutdown –h now

4 Créer un snapshot popc_clean

Annexe 2 : schéma installation hepia



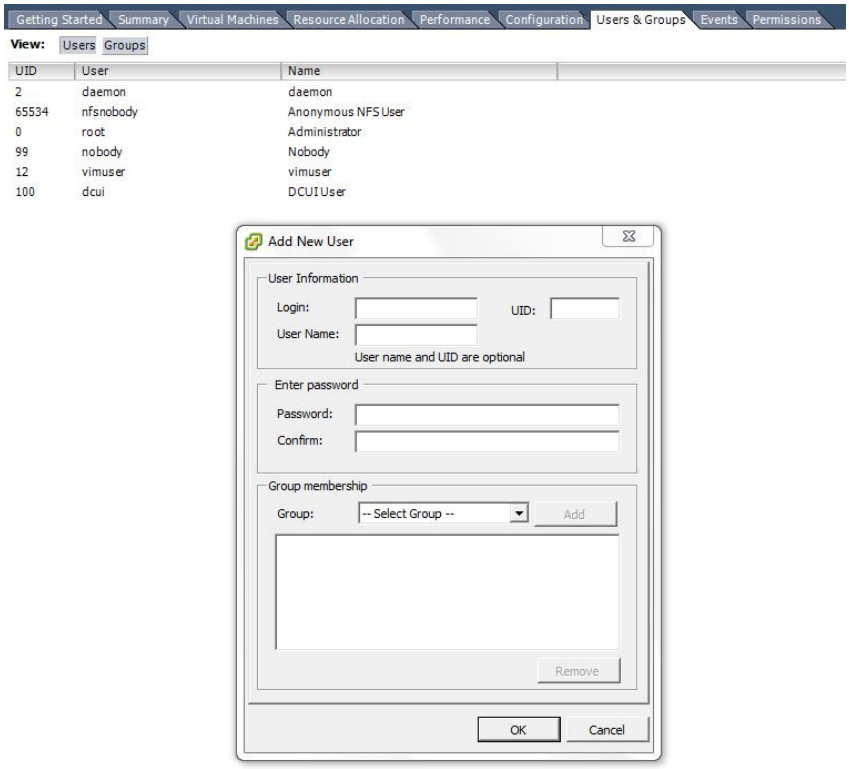


Annexe 3 : Compte utilisateur avec droits restreints

1. Création d'un utilisateur

Créer un nouvel utilisateur avec les permissions par défaut.

- Dans vSphere Client, se positionner sur l'hôte ESXi.
- Onglet Users & Groups (View: Users)
- Clic droit -> Add...



Il y a 3 champs obligatoires à remplir:

- Login: Nom utilisé pour se connecter
- Password: Le mot de passe utilisé pour se connecter
- Confirm: Une confirmation du mot de passe

Remarque: Le password doit respecter certains critères sans quoi on obtient une erreur de la part de l'ESXi. Apparemment les critères ont été revus à la hausse au passage de ESXi 3.5 -> 4.0. Par défaut, il faut:

- un password de 8 caractères minimum pouvant comprendre des minuscules, majuscules, digits, caractères spéciaux.
- Il ne doit pas contenir le "login".

Utilisateur créé pour le test:

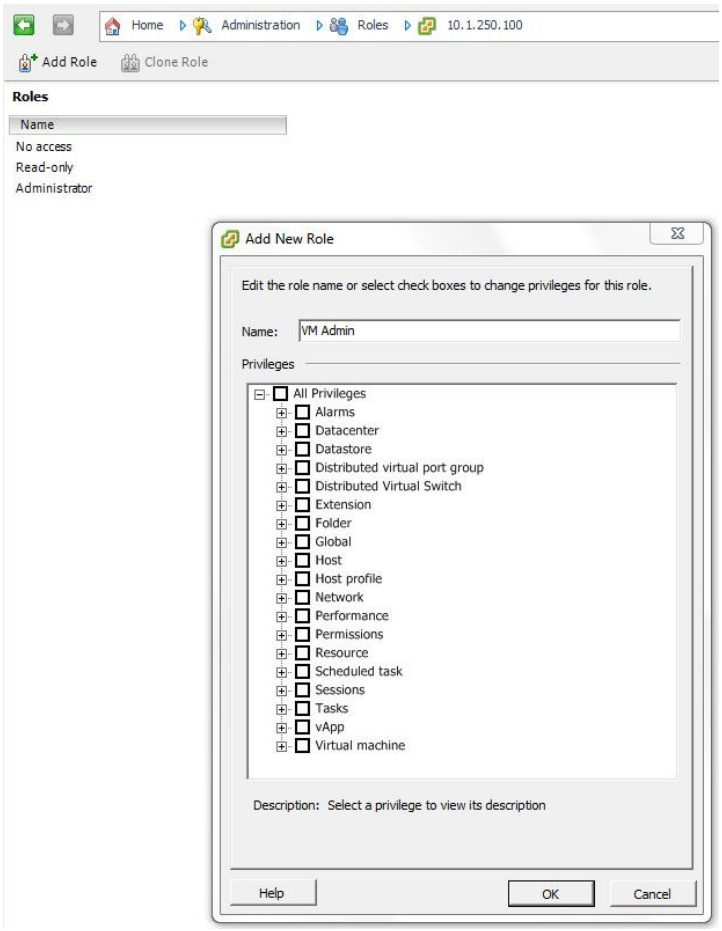
Login: vpopc
User name: VPOPC User
Password: VISAG2010
Group: Par défaut l'utilisateur est placé dans le groupe "users"

2. Création d'un rôle

Les rôles servent à regrouper un ensemble de permissions déterminées pour les assignées plus facilement à un utilisateur.

Nous allons donc créer un rôle (VM Admin).

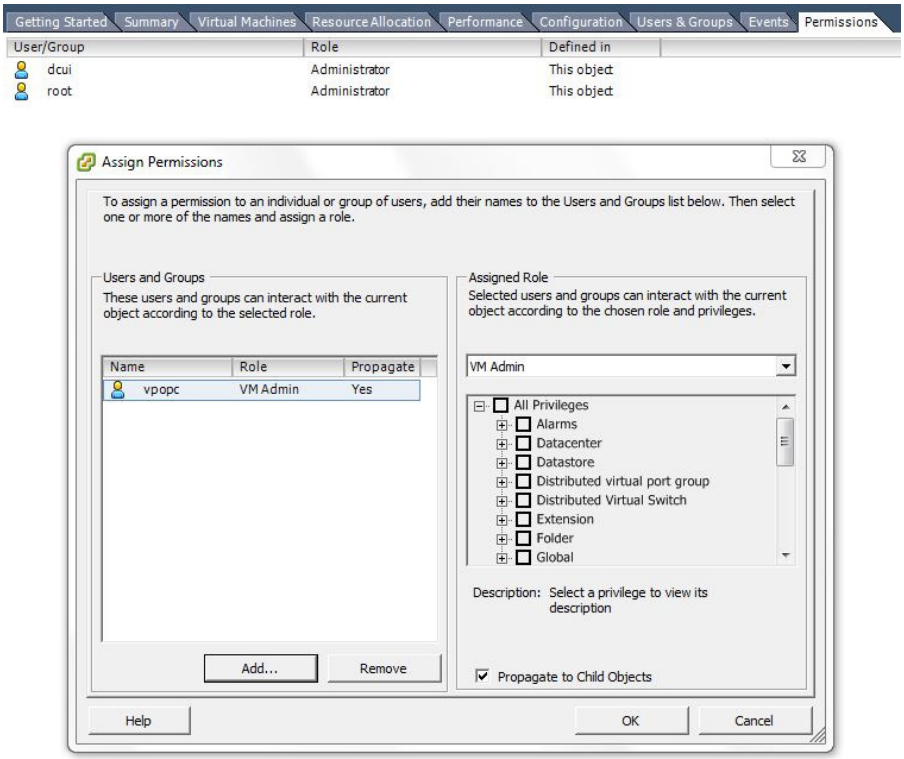
Home -> Roles -> Add Role



Pour commencer, nous laisserons ce rôle sans aucune permission et elles seront ajoutées au fur et à mesure lors des tests des fonctions que la VM Admin utilisera.

Il faut ensuite associer ce nouveau rôle que nous venons de créer avec l'utilisateur qui a été créé en §1 (vpopc).

Se rendre dans Home > Inventory puis sous la vue hôte ESXi, aller sous l'onglet "Permissions".



Clic droit -> Add permission...

Dans la partie de droite, il faut sélectionner le rôle que nous venons de créer.

Dans la partie de gauche, il faut ajouter la liste de tous les utilisateurs ESXi qui peuvent disposer des permissions définies dans notre rôle (VM Admin), un seul utilisateur (vpopc) dans notre cas.

3. Tests & ajouts permissions

Pour tester les différentes fonctions que la VM Admin utilise sur notre hyperviseur, j'ai utilisé l'outil fourni par Fribourg qui se nomme « vpopcwrapper ».

La première chose à faire, est de contrôler quelles sont les actions que nous pouvons effectuer avec le compte utilisateur « vpopc » en laissant les droits par défaut (c'est-à-dire sans en avoir ajouté dans notre rôle).

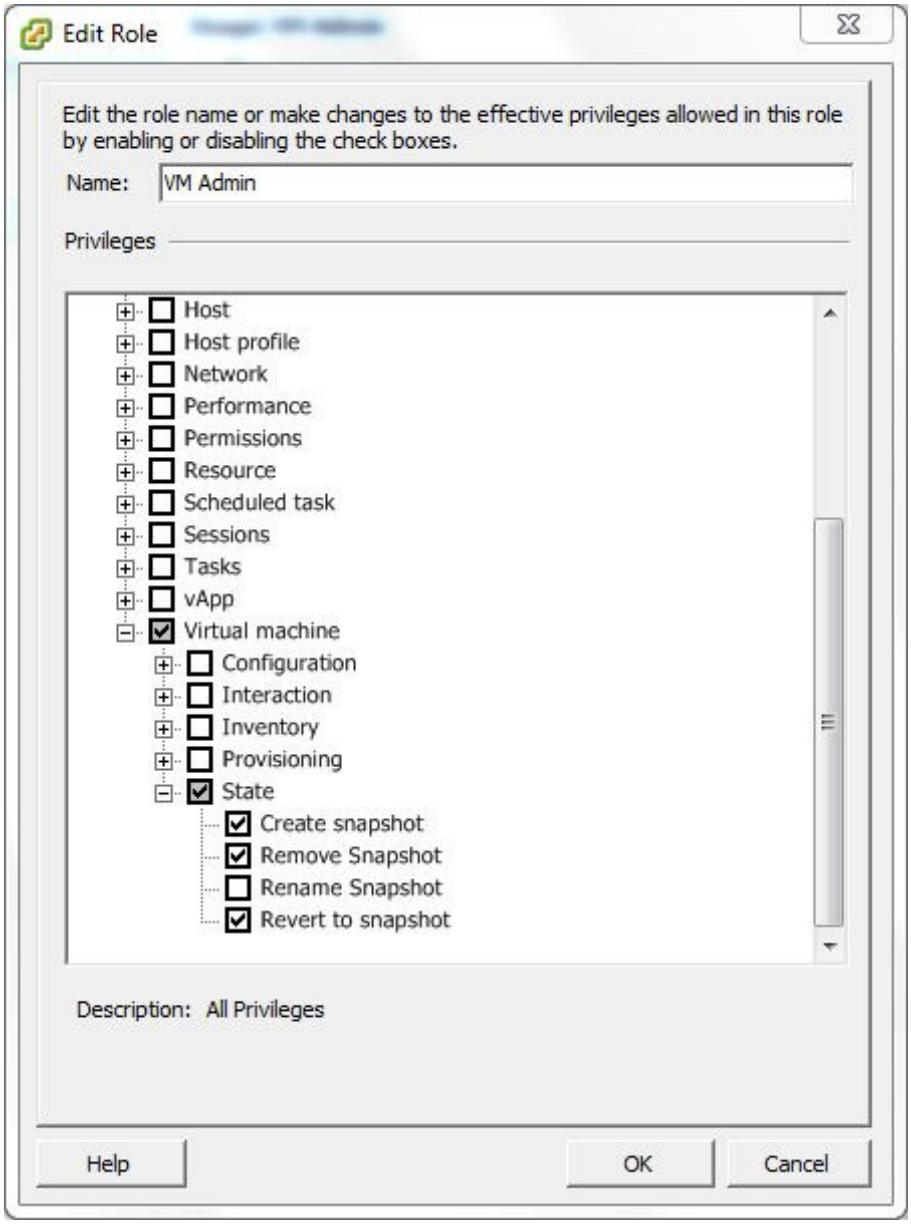
Voici les fonctions du vpopcwrapper à tester :

- **Connexion à un hyperviseur**
- **Connexion à une vm**
- **Récupérer l'IP d'une VM**
- **Récupérer la MAC d'une VM**
- **Récupérer l'état d'une VM**
- **Récupérer la mémoire libre de l'hyperviseur**
- **Récupérer diverses informations sur l'hyperviseur**
- **Récupérer le nombre de snapshots d'une VM**
- **Récupérer la liste des snapshots d'une VM**
- **Revert snapshot**
- **Créer un snapshot**
- **Supprimer un snapshot**
- **Récupérer la mémoire utilisé par une VM**
- **Définir la mémoire d'une VM**
- **Allumer une VM**
- **Eteindre une VM**

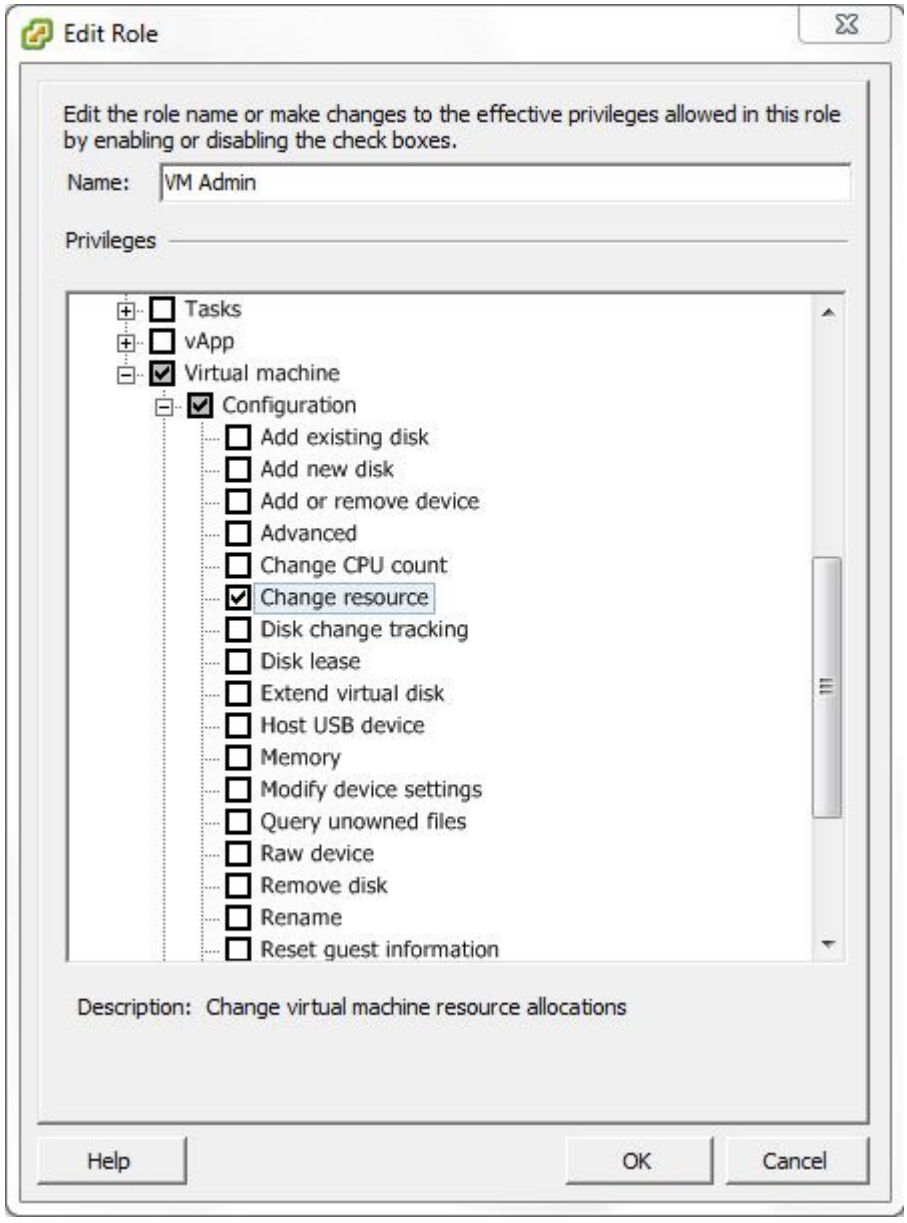
Après les avoir testé une par une, voici celles qui se passent de droits supplémentaires :

- **Connexion à un hyperviseur**
- **Connexion à une vm**
- **Récupérer l'IP d'une VM**
- **Récupérer la MAC d'une VM**
- **Récupérer l'état d'une VM**
- **Récupérer la mémoire libre de l'hyperviseur**
- **Récupérer diverses informations sur l'hyperviseur**
- **Récupérer le nombre de snapshots d'une VM**
- **Récupérer la liste des snapshots d'une VM**
- **Récupérer la mémoire utilisé par une VM**

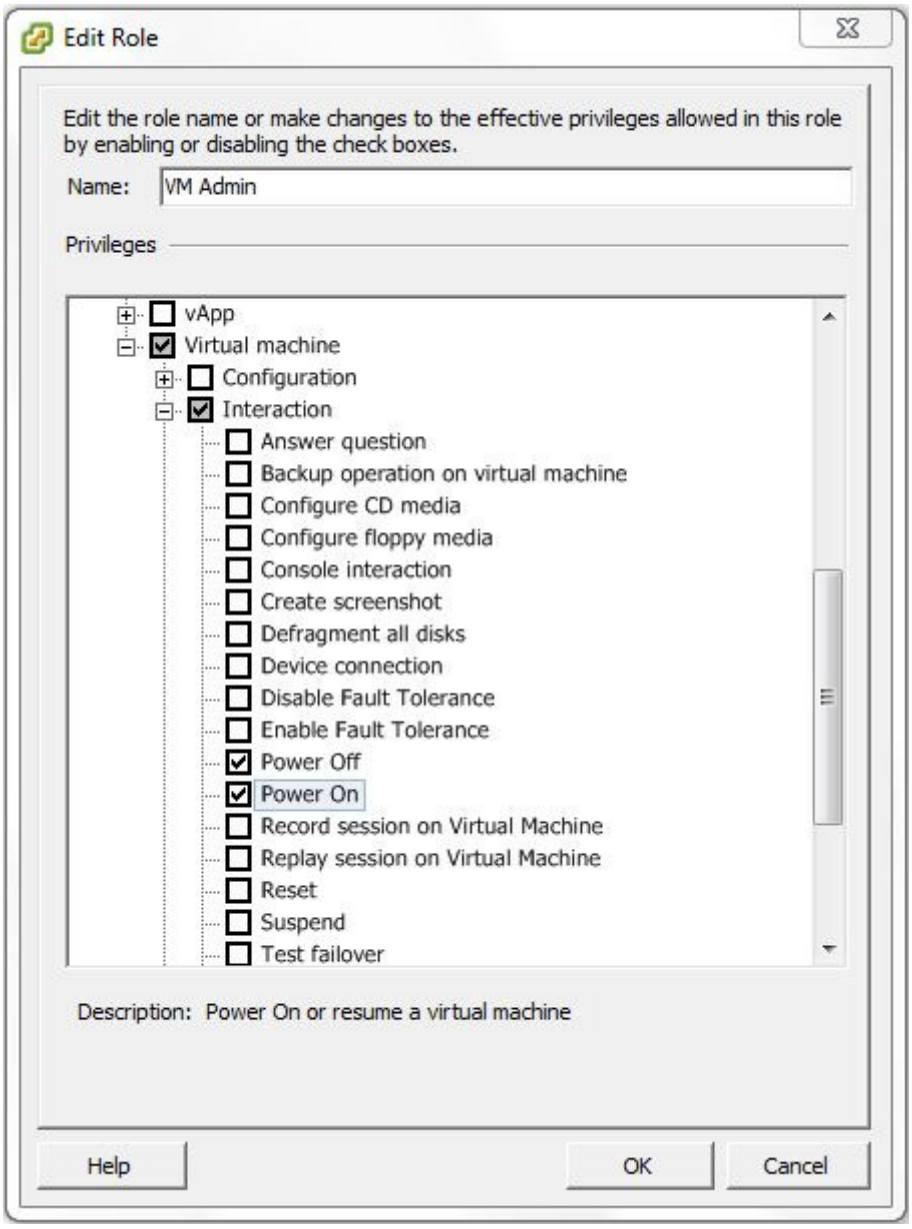
Voici les permissions à ajouter au rôle VM Admin pour **les fonctions de snapshot** :



Les permissions à ajouter pour **la définition de la mémoire d'une VM** :

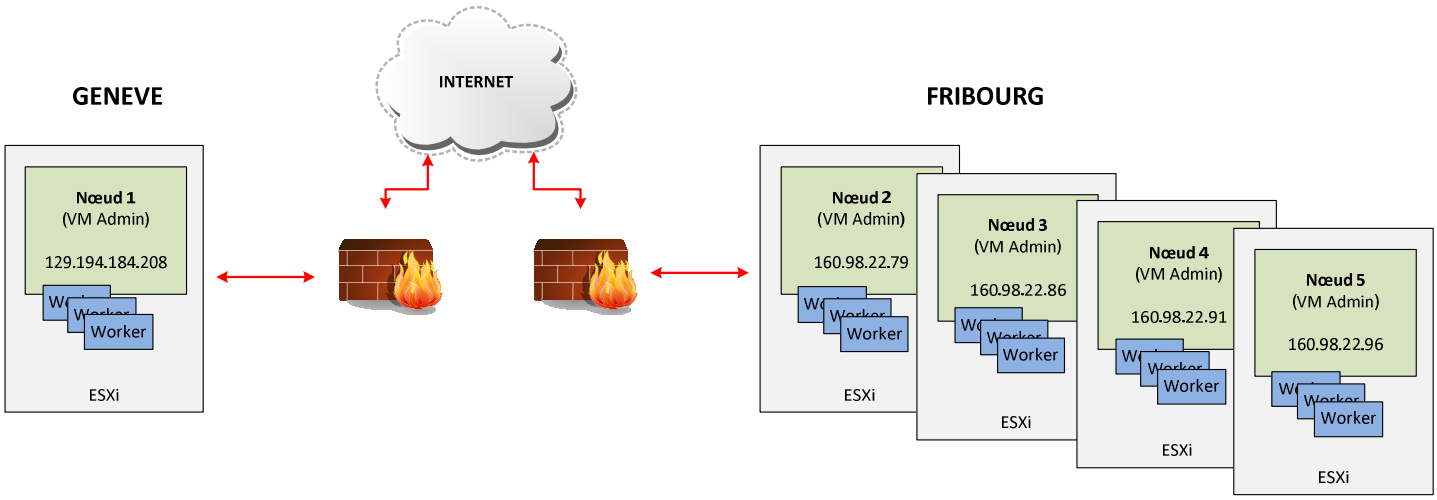


Les permissions à ajouter pour **allumer et éteindre une VM** :


Edit Role
Edit the role name or make changes to the effective privileges allowed in this role by enabling or disabling the check boxes.
Name:
VM Admin
Privileges
vApp
Virtual machine
Configuration
Interaction
Answer question
Backup operation on virtual machine
Configure CD media
Configure floppy media
Console interaction
Create screenshot
Defragment all disks
Device connection
Disable Fault Tolerance
Enable Fault Tolerance
Power Off
Power On
Record session on Virtual Machine
Replay session on Virtual Machine
Reset
Suspend
Test failover
Description: Power On or resume a virtual machine
Help
OK
Cancel

Annexe 4 : Analyse de la démo entre Genève et Fribourg

Schéma :



Déroulement du scénario :

Pour réaliser ce test entre les deux sites Genève et Fribourg, nous avons utilisé un programme de démo mis au point par Fribourg et qui consiste simplement à créer autant d'objets que l'on souhaite sur notre réseau et à s'échanger des ID. Le but n'étant pas ici de tester les performances de calcul mais d'avoir des données concrètes pour un test à travers des firewalls et internet.

Pour ce qui est de l'exécution de ce scénario, voici comment nous avons procéder :

1. Démarrer POPC respectivement sur les nœuds 1, 2, 3, 4, 5 :

```
> SXXpopc start
```

2. Lancer le programme de démo depuis le nœud 2 :

```
> cd popcpp_.../demos/demopopc
> popcrun obj.map ./main 5 - - - - -
```

Pour ce test, nous lançons la démo avec 5 comme paramètre ce qui va créer 5 objets dont 1 sur chaque nœud (Worker).

Nous obtenons l'exécution suivante :

```
START of ./main program with 5 objects
POPCobject with ID=1 created (by JobMgr) on machine:160.98.22.80
POPCobject with ID=2 created (by JobMgr) on machine:160.98.22.83
POPCobject with ID=3 created (by JobMgr) on machine:160.98.22.87
POPCobject with ID=4 created (by JobMgr) on machine:160.98.22.84
POPCobject with ID=5 created (by JobMgr) on machine:129.194.184.208
POPCobject:1 on machine:160.98.22.80 is sending his iD to object:2
POPCobject:2 on machine:160.98.22.83 is receiving id =1
POPCobject:2 on machine:160.98.22.83 is sending his iD to object:3
POPCobject:3 on machine:160.98.22.87 is receiving id =2
POPCobject:3 on machine:160.98.22.87 is sending his iD to object:4
POPCobject:4 on machine:160.98.22.84 is receiving id =3
POPCobject:5 on machine:129.194.184.208 is sending his iD to object:1
POPCobject:1 on machine:160.98.22.80 is receiving id =5
POPCobject:5 on machine:129.194.184.208 is waiting 2 sec

END of ./main program
```

Tout se déroule sans erreur et la communication entre les objets se fait normalement à travers les firewalls et internet.

Analyse des échanges depuis le site de Genève :

Nous utilisons Wireshark pour cette analyse des flux.
La capture est disponible ici : http://www.tdeig.ch/visag/demo2/capture.pcap

Les échanges sont bien encapsulés avec SSH ; voir figures ci-dessous

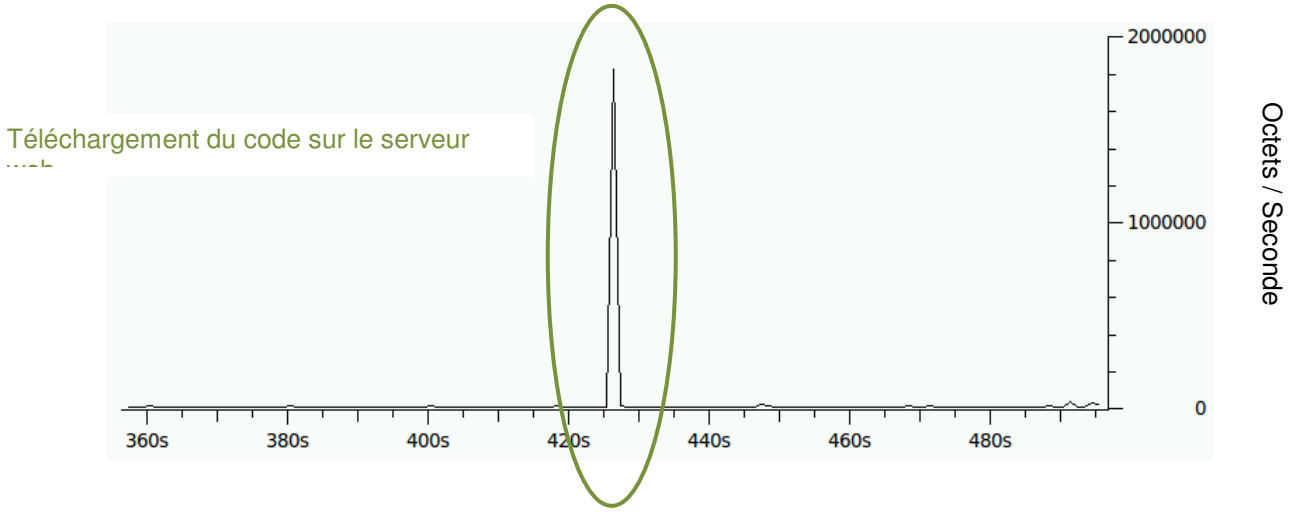
Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	En
Frame	100.00 %	4069	2182881	0.035	0	
Ethernet	100.00 %	4069	2182881	0.035	0	
Internet Protocol	100.00 %	4069	2182881	0.035	0	
Transmission Control Protocol	97.94 %	3985	2174447	0.035	2727	
SSH Protocol	30.84 %	1255	310866	0.005	1255	
Hypertext Transfer Protocol	0.05 %	2	1339	0.000	1	
Line-based text data	0.02 %	1	1139	0.000	1	
Data	0.02 %	1	134	0.000	1	
User Datagram Protocol	2.02 %	82	8310	0.000	0	
Domain Name Service	1.77 %	72	6906	0.000	72	
Bootstrap Protocol	0.05 %	2	684	0.000	2	
Network Time Protocol	0.20 %	8	720	0.000	8	
Internet Control Message Protocol	0.05 %	2	124	0.000	2	

Address A	Port A	Address B	Port B	Packets	Bytes
129.194.184.208	49093	160.98.8.50	http	1973	1815201
160.98.22.79	59767	129.194.184.201	ssh	150	27704
129.194.184.201	60954	160.98.22.79	ssh	93	14232
160.98.22.79	51817	129.194.184.208	ssh	63	9348
160.98.22.79	59756	129.194.184.201	ssh	57	8500
160.98.22.79	59758	129.194.184.201	ssh	55	8242
160.98.22.84	34513	129.194.184.208	ssh	55	8332
160.98.22.79	59989	129.194.184.201	ssh	54	8230
160.98.22.79	59994	129.194.184.201	ssh	54	8470
160.98.22.79	60001	129.194.184.201	ssh	54	8394
160.98.22.79	60022	129.194.184.201	ssh	54	8656
160.98.22.79	59765	129.194.184.201	ssh	53	8128
160.98.22.79	59769	129.194.184.201	ssh	52	8062
160.98.22.79	59991	129.194.184.201	ssh	52	8334
129.194.184.208	60269	129.194.184.201	ssh	47	8300
129.194.184.201	38593	129.194.184.208	ssh	46	8410
129.194.184.201	60948	160.98.22.79	ssh	45	8120
129.194.184.208	60271	129.194.184.201	ssh	45	8232
129.194.184.208	40382	160.98.22.80	ssh	45	8216
129.194.184.201	60957	160.98.22.79	ssh	43	7988
129.194.184.201	60952	160.98.22.79	ssh	42	7922
129.194.184.201	41832	160.98.22.79	ssh	42	8402
129.194.184.201	36154	160.98.22.79	ssh	41	8544
129.194.184.201	36171	160.98.22.79	ssh	41	7968
129.194.184.208	48147	160.98.22.79	ssh	41	7760
129.194.184.201	36144	160.98.22.79	ssh	40	8478
160.98.22.79	51813	129.194.184.208	ssh	40	6836
129.194.184.201	36164	160.98.22.79	ssh	39	8412
129.194.184.208	48131	160.98.22.79	ssh	39	7612
129.194.184.208	48150	160.98.22.79	ssh	39	7628
129.194.184.208	48152	160.98.22.79	ssh	39	7612

La première ligne (protocole http) de la deuxième capture mérite une explication.

- Il ne s'agit pas de communication entre nœuds.
- Ce flux correspond au téléchargement du code à exécuter depuis un serveur web de Fribourg vers la VM Admin du nœud 1

Au niveau de la charge réseau, nous constatons que ce téléchargement http génère le volume le plus important :



Et pour terminer, quelques statistiques sur cette capture.

Traffic	Captured	Displayed	Marked
Packets	4069	4069	0
Between first and last packet	495.500 sec		
Avg. packets/sec	8.212		
Avg. packet size	536.466 bytes		
Bytes	2182881		
Avg. bytes/sec	4405.411		
Avg. MBit/sec	0.035		