Cipher

Milan Soragna Aurèle Dunand Thomas Boussit

Table des matières

Scripts

Notre projet va être composé de nombreux scripts, qui permettent chacun de générer une partie de l'infrastructure.

Nos scripts sont tous disponibles sur un répertoire *GitHub*, ce qui est pratique pour nous pour travailler en simultané et avoir accès aux scripts, mais sera aussi éventuellement utile plus tard pour automatiquement télécharger les scripts à distance.

DHCP

Pour le *DHCP*, nous avons pour le moment écrit 2 scripts, un qui doit être exécuté sur le serveur afin de le mettre en place, et un autre qui devra être utilisé sur la machine cliente afin de modifier les paramètres de la machine pour qu'elle utilise le serveur *DHCP*.

Voici le lien vers ces deux scripts :
https://github.com/AngarosGamer/SAE4/tree/main/dhcp

Pare-feu

Pour le pare-feu, il s'agit cette fois d'une combinaison de fichiers qui vont tous être exécutés sur le serveur / routeur agissant comme pare-feu.

En l'occurrence, un script très basique sert à remettre à jour le pare-feu et toutes les règles qui le compose. C'est pratique en cas de modification du pare-feu où l'on veut repartir d'une nouvelle base.

Un autre script va gérer les règles relatives aux paquets sortant du routeur, avec des règles fonctionnant sur la base d'une whitelist.

Un script très similaire va gérer les règles relatives aux paquets passant par le routeur, avec des règles en whitelist. C'est ce script par exemple qui va gérer les règles individuelles de chaque serveur et machine pour autoriser ou bloquer la connexion.

Un dernier script permet l'installation de *nftables*, et de lancer les 3 scripts précédents en même temps pour entièrement remettre à jour le pare-feu.

Le lien vers ces quatre scripts est :
https://github.com/AngarosGamer/SAE4/tree/main/firewall

Il faut noter que ces scripts ne sont pas terminés, puisque les IP des serveurs ne sont pas encore statiques, et que nous prévoyons d'ajouter d'autres scripts et fonctionnalités. Les versions actuelles sont préliminaires.

Serveurs Web

Pour que notre infrastructure soit complète, il faut mettre en place un intranet ainsi qu'un extranet. A cet effet, nous avons écrit un script qui se charge d'installer *Apache* et *PHP8.2* sur la machine qui l'exécute, et qui met en place les fichiers de l'intranet.

Voici le lien vers ce script et la base de l'intranet :
https://github.com/AngarosGamer/SAE4/tree/main/intranet

Cette mise en place est aussi une version préliminaire est n'est pas la version finale. Nous allons inclure des changements dans les scripts d'installation, et dans l'intranet lui-même.

Journalisation

Pour la journalisation, nous avons mis en place deux scripts. Tout comme pour le *DHCP*, un script va devoir s'exécuter sur le serveur afin de mettre en place les différents services relatifs à la réception des journaux et leur classement, et un autre script doit être utilisé sur la machine cliente pour correctement envoyer les journaux sur le serveur de logs.

Ces scripts sont disponibles ici :
https://github.com/AngarosGamer/SAE4/tree/main/journalisation

Les scripts de la journalisation ne sont pas terminés et représentent un travail qui n'est pas abouti, car le serveur réceptionnant les logs n'est pas installé et n'a pas encore son adresse fixe.

Kerberos

Kerberos représente un pilier de notre infrastructure, qui va devoir sécuriser notamment les connexions *NFS*, et gérer l'authentification d'autres services de l'infrastructure.

Nous avons cette fois aussi 2 scripts, un sur la machine serveur, qui sert pour le moment juste à installer *Kerberos*, et l'autre est un script sur les machines clientes, qui est aussi utilisé simplement pour l'installation.

Cela étant dit, nous avons aussi mis en place 2 fichiers de configuration *Kerberos* (*KDC* et *KRB5*), avec une partie des paramètres importants pour la mise en place du serveur *Kerberos*.

Ces fichiers sont disponibles avec le lien :
https://github.com/AngarosGamer/SAE4/tree/main/kerberos

LDAP

Nous avons entrepris la mise en place des services *LDAP* relativement tôt car nous savons que c'est une tâche relativement compliquée et nous voulions prendre de l'avance sur sa mise en place. A cet effet, nous avons écrit 3 scripts :

Une installation du serveur *LDAP* et de sa configuration est prévue dans un script, qui permettra d'automatiser tout le processus d'installation de la machine serveur.

De la même manière, un script a aussi été créé pour mettre en place la machine cliente avec le bon serveur *LDAP*. Ce script n'est pas entièrement automatisé dans la mesure où il est nécessaire de renseigner le nom du nouvel utilisateur, son *uid* et son *gid* en paramètre.

Dernièrement, nous avons la base d'un fichier permettant l'ajout d'un nouvel utilisateur au service *LDAP*, et qui permet d'ajouter manuellement un nouvel utilisateur.

Pour faciliter l'insertion des informations dans l'annuaire LDAP, nous avons prévu d'utiliser *phpldapAdmin* ce qui nous permettra de mieux le visualiser, nous pourrions prévoir sa configuration en script afin de le proposer à l'administrateur qui utilisera le système mais cela n'est pas encore défini.

Ces trois scripts sont disponibles ici :
https://github.com/AngarosGamer/SAE4/tree/main/ldap

Le service *LDAP*, bien qu'il soit bien avancé, n'est pas encore opérationnel. Ces scripts sont donc encore en cours de production et vont subir des modifications avant la fin de notre projet.

NFS

Le service *NFS* permet d'avoir un serveur centralisé qui contient (dans notre cas au moins) tous les répertoires des utilisateurs. Ceci permet à un individu, peu importe la machine utilisée, de retrouver ses répertoires et fichiers.

Pour le mettre en place, nous avons ici aussi créé 3 scripts :

Un script permettant l'installation et la mise en opération du service *NFS* sur le serveur, qui permet l'autoconfiguration du service.

Un script permettant l'installation sur la machine cliente, qui permet de répertorier le serveur sur lequel il faut chercher les documents, et modifier la configuration de la machine pour chercher ce répertoire dès le login d'un utilisateur.

Un dernier script permet d'ajouter une nouvelle machine au service *NFS*, puisque le service *NFS* doit répertorier les adresses IP auxquelles il peut répondre. Ce script devra être exécuté sur le service pour ajouter une nouvelle machine qui doit utiliser le service *NFS*. Ce script n'est pas entièrement automatisé dans la mesure où il est nécessaire de renseigner l'adresse IP de la machine cliente devant être ajoutée en paramètre.

Les scripts sont disponibles ici :
https://github.com/AngarosGamer/SAE4/tree/main/nfs

Le service *NFS* est fonctionnel dans son état actuel, et son utilisation devrait permettre au serveur et machine cliente de communiquer normalement. Cela dit, il se peut que les fichiers soient modifiés dans le futur afin d'apporter d'autres fonctionnalités.

DNS

Les serveurs *DNS bind9* représentent une partie structurante de notre architecture.

Les scripts, pour le moment embryonnaires pour les serveurs *DNS* de la *DMZ* et de l'intranet, permettent néanmoins d'avoir une idée des scripts dans leur forme finale.

Les scripts *DNS* pour la DMZ et ceux de l'intranet possèdent une grande partie commune. Ils permettent de mettre en place, pour l'intranet et pour la *DMZ*, un serveur, et permettent aux ordinateurs externes d'accéder aux ressources internes et aux ordinateurs de l'intranet d'accéder à des ressources externes.

Voici le lien vers ces deux scripts :
https://github.com/AngarosGamer/SAE4/tree/main/dns

Serveur SGBD

Cette partie a nécessité l'ajout d'un script simple qui installe le serveur de base de données *Postgres*. C'est un script qui va devoir s'exécuter sur le serveur, qui va installer tous les packages et logiciels de mise en place du serveur SGBD.

Le script est disponible à cette adresse :
https://github.com/AngarosGamer/SAE4/tree/main/sgbd

Zabbix

Pour la surveillance du réseau, nous avions choisi le logiciel Zabbix, pour lequel nous avons écrit le script qu'il faudra exécuter sur la machine serveur pour le mettre en place. Ce script automatise toute la mise en place du service et la configuration, tout en laissant aux utilisateurs le choix sur certains paramètres.

Le script est accessible ici :
https://github.com/AngarosGamer/SAE4/tree/main/zabbix

Le script est entièrement fonctionnel et ne nécessitera normalement pas d'autres modifications ultérieures.

Installation Machines

Certains scripts ne rentrent pas dans une catégorie en particulier, donc nous en parlons brièvement dans cette partie.

Pour l'installation des machines virtuelles, nous avons décidé d'écrire des scripts permettant l'installation automatique d'une machine.

Il y a 4 scripts (dont fichiers de configuration) utiles :

Un fichier *preseed*, qui permet de renseigner à Debian toutes les informations à savoir pour l'installation. Ce fichier permet de faire une installation Debian sans aucune manipulation utilisateur. Ce fichier n'a pas vocation à subir plus de modifications à l'heure actuelle, il semble terminé.

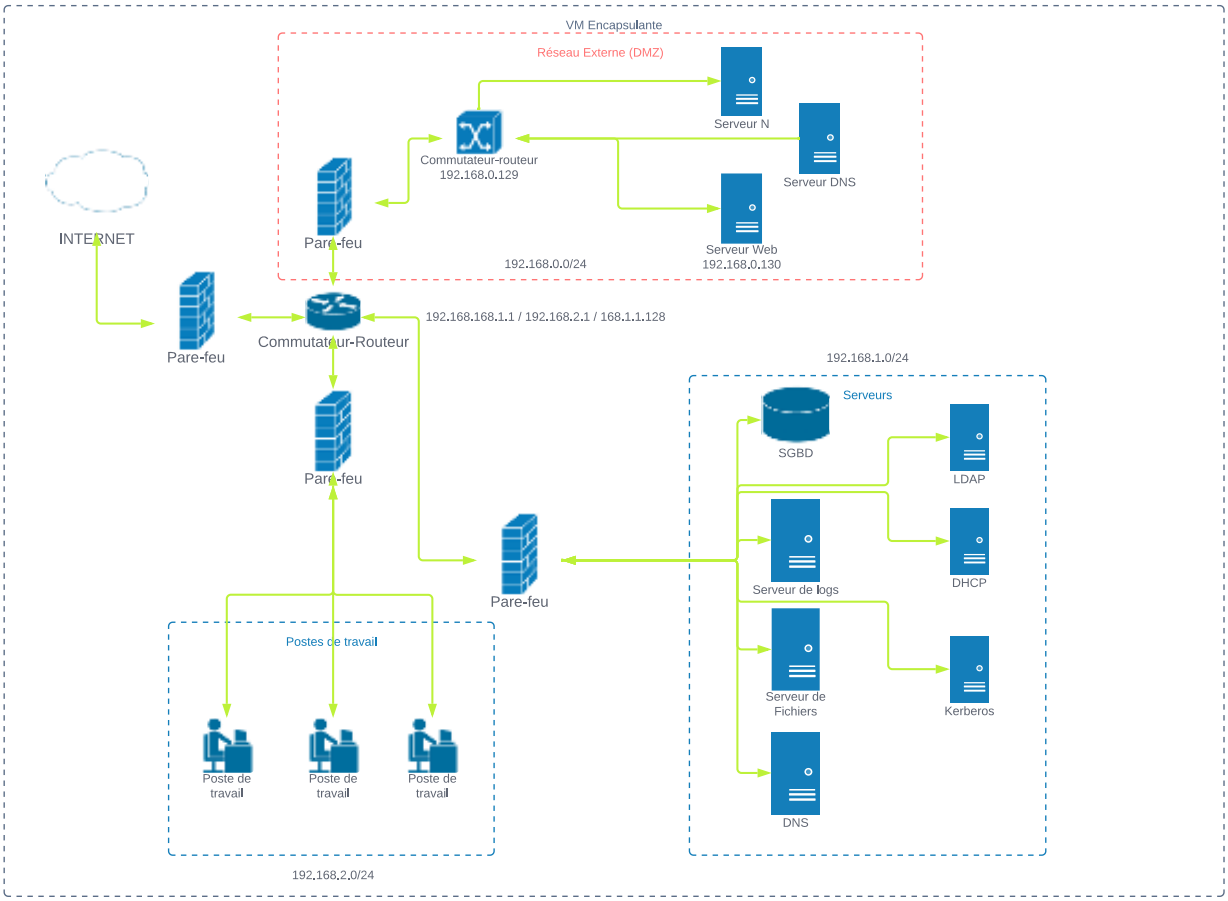
Un script qui va construire, à partir du fichier *preseed* d'une part, et d'un ISO debian de l'autre, une nouvelle image ISO Debian intégrant le fichier preseed. Ce script ne va pas être modifié, et n'a en théorie pas de raisons d'être exécuté plus d'une fois. Après son utilisation, il sera possible d'utiliser l'ISO en sortie pour la création de multiples VM. Nous avons pensé à l'intégrer ici afin de laisser le choix à l'utilisateur d'éventuellement créer sa propre image ISO faisant utilisation de ses propres paramètres.

Un script va ensuite utiliser virt-install afin d'aller récupérer les paramètres entrés par l'utilisateur et l'ISO *preseed* pour lancer toute l'installation automatiquement.

Dernièrement, un script post-installation permettra de remettre toute la configuration par défaut d'une machine. Notamment, il change le hostname de la machine, efface

l'utilisateur par défaut, en créé un nouveau, et modifie le mot de passe du compte root. Ce script peut être amené à évoluer, par exemple pour automatiquement télécharger et lancer les scripts pour NFS, LDAP, … et modifier d'autres paramètres.

Organisation de l'infrastructure



Voici le script représentant l'infrastructure actuelle telle que nous prévoyons de la mettre en place.

Les machines spécifiques ayant une adresse IP sont celles qui sont pour le moment installées. Ainsi, nous avons installé les machines :

- Routeur Principal : 192.168.1.1 / 192.168.2.1 / 192.168.0.128
- Serveur SGBD : 192.168.1.2
- Routeur de DMZ : 192.168.0.129
- Serveur Web (DMZ) : 192.168.0.130

Nous avons 3 réseaux virtuels connectés par un routeur principal, ainsi qu'un réseau NAT, qui fonctionnent sous la forme :

- Default : Le réseau NAT qui passe les connexions internes à la VM vers le monde extérieur.
    - Réseau 192.168.122.0/24
    - Machines allant de 192.168.122.2 à 192.168.122.254
- DMZ : Le réseau virtuel de la DMZ qui lie toutes les machines faisant partie de la DMZ

    - Réseau 192.168.0.0/24
    - Machines allant de 192.168.0.128 à 192.168.0.254
- Serveurs : Le réseau virtuels englobant les serveurs de l'infrastructure
    - Réseau 192.168.1.0/24
    - Machines allant de 192.168.1.1 à 192.168.1.254
- Machines : Le réseau englobant les postes de travail de l'intranet
    - Réseau 192.168.2.0/24
    - Machines allant de 192.168.2.1 à 192.168.2.254

Machines de l'infrastructure

Pour notre infrastructure, nous avons choisi de mettre en place 13 machines différents pour la démonstration, bien que ce que nous avons mis en place pourrait accommoder un plus grand nombre de machines.

Les machines sont séparées en plusieurs catégories :

- Les commutateurs-routeurs

Il y aura 2 routeurs dans notre infrastructure :

  - Le routeur principal, ainsi que le service *Zabbix*
  - Le commutateur de la DMZ

Dans notre cas, nous avons décidé de ne pas mettre un commutateur-routeur dans le réseau de l'intranet (postes de travail et serveurs). Bien que dans la réalité cela puisse être nécessaire (notamment dans le cas où le commutateur-routeur principal n'a pas assez de ports libres), mais ce problème n'a pas d'impact puisque nous travaillons avec des machines virtuelles.

- Les serveurs

Nous aurons 9 serveurs dans notre infrastructure, chacun remplira son propre service. Dans la réalité, selon le budget disponible, il aurait été possible de configurer plusieurs services sur un ou plusieurs serveurs afin de réduire le nombre effectif de machines nécessaires.

  - Un serveur *DNS* dans la DMZ
  - Un serveur web « extranet » dans la DMZ
  - Un serveur de Base de données dans l'intranet
  - Un serveur *LDAP* dans l'intranet
  - Un serveur de logs dans l'intranet
  - Un serveur *DHCP* dans l'intranet
  - Un serveur de fichiers dans l'intranet
  - Un serveur *Kerberos* dans l'intranet
  - Un serveur *DNS* dans l'intranet

Chacun de ces serveurs n'exécute que la tâche qui lui est propre, comme nous fonctionnons avec des VM, chaque machine a une tâche dédiée.

- Les postes de travail

Pour la démonstration de notre infrastructure, nous allons mettre en place 2 postes de travail, mais il est possible, via les scripts d'installation, de rapidement configurer une nouvelle machine qui prendra sa place.

Les deux machines seront les mêmes en termes de configuration, et servent à l'organisation de machines sur lesquelles les employés et administrateurs peuvent travailler. Ces machines auront une interface graphique, quelques logiciels de base

pour le développement, et bien sûr la configuration requise pour les rendre « compatible » avec notre infrastructure.

Contenu des machines

Pour le moment, nous venons de mettre en place les machines virtuelles. C'est pourquoi les machines sont toutes vides et n'ont pas de contenu intégré. Les scripts d'installation et configuration sont tous hébergés sur GitHub et lorsque nous serons prêts à lancer l'infrastructure nous installerons et exécuterons ces scripts.

Nos tests sont faits sur des machines virtuelles temporaires et nous utiliserons le système mis en place pour installer les logiciels et services en une fois comme une situation réelle.