AmarOk

Retour d'expérience, l'installation de Jitsi Meet

11 mai 2019

Table des matières

Pour le CHATONS en gestation du coin (https ://facil.services/), je donne un coup de patte depuis quelques semaines à mettre des services en place. En ce moment, je regarde pour héberger une instance Jitsi meet sur un des serveurs disponibles. L'instance n'est pas encore ouverte, mais il ne reste plus grand chose à faire. Voici donc un petit billet retour d'expérience sur l'hébergement de ce service.

1. Première étape, l'installation manuelle

Pour la petite histoire, la première tentative d'hébergement du service fonctionnait bien pendant un petit moment, mais a complètement arrêté de fonctionner soit après la migration vers le serveur final soit pendant la mise en place des outils de back-up ou monitoring.

La première chose que j'ai donc décidé de faire était de l'installer sur une machine virtuelle en mode bridge répliquant la configuration du serveur (Debian Stretch, 1 Go de RAM (2 à la base), 2 coeurs (bon sur la vraie machine c'est un Xeon)) et de suivre les instructions de cette page : https ://github.com/jitsi/jitsi-meet/blob/master/doc/quick-install.md

La seule vraie grosse différence étant que cette VM était dans mon réseau local, avec un certificat auto-signé au lieu du certificat Let's Encrypt habituel, cependant, on a pu voir que le Jitsi fonctionnait correctement et faire une conférence à plusieurs dessus.

Comme le tout avait l'air de fonctionner, j'ai réalisé exactement les mêmes étapes (avec le certificat auto-signé) sur le serveur qui nous servira pour la production (serveur remis à zéro). Une fois qu'on a pu voir que l'instance fonctionnait aussi sur ce serveur, la prochaine étape était de remettre les outils classiques d'un serveur, à savoir la mise en place des back-ups, du monitoring et de l'automatisation de l'installation, et la gestion des certificats.

2. Début de l'automatisation

Pour la mise en place des serveurs, Ansible est utilisé. Pour éviter de ré-inventer la roue, on utilise des rôles déjà utilisés par d'autres personnes. Pour Jitsi, j'ai trouvé ce projet : https ://github.com/freedomofpress/ansible-role-jitsi-meet

Même si ce rôle n'a pas été mis à jour depuis 2017, il exite une pull request datant du 28 Mars (https ://github.com/freedomofpress/ansible-role-jitsi-meet/pull/43) Je décide donc de partir sur ce rôle patché. Et je fais mon petit playbook `jitsi.yml` :

```
```

Je lance mon petit playbook et... bam let's encrypt ne marche pas, fin de l'installation.

3. Débogage

Premier problème, le challenge pour le certificat échoue et donc le rôle `thefinn93.letsencrypt` échoue. Au final je me rends compte que le `nginx` de la machine n'écoutait que sur IPv4, donc le challenge en IPv6 ne passait forcément pas.

Je relance, les challenges passent, mon certificat est signé, passage à Jitsi, tout à l'air de bien passer, reload de `nginx`, BAM il ne passe pas. Et le serveur ne répond pas en https. Je me connecte à la machine, je ne comprends pas vraiment pourquoi `nginx` n'est pas relancé, mais comme il s'agit de la fin du rôle, je décide de relancer à la main (`service nginx restart`) pour voir si Jitsi fonctionne au moins.

Super, l'interface est bien présente, je me connecte, la caméra s'affiche, je demande à une autre personne dans la pièce de se connecter à l'adresse de la conférence, crash de la conférence

Je me dis que le problème vient peut-être du rôle, je décide de purger à la main et de relancer. Petite note, `jitsi-meet-web-config` se désinstalle mal lorsqu'on purge *nginx* avant *Jitsi*. Pour fixer la désinstallation, il faut enlever les lignes suivantes du fichier `/var/lib/dpkg/info/jitsi-meet-web-config.postrm` :

```
1        if [ -x "/etc/init.d/nginx" ]; then
2            invoke-rc.d nginx reload
3        fi
```

Bref, cette fois le rôle se passe parfaitement, je retente de connecter deux ordinateurs à l'instance, et... crash.

Bon, il est temps de checker les logs. Pour ces logs, jitsi possède au moins deux fichiers qui sont généralement intéressants :

— `/var/log/jitsi/jicofo.log` : qui m'a permis de trouver un petit problème de ssl pendant que j'utilisais mon certificat auto-signé. Ce premier me dit à ce moment qu'il ne trouve pas de bridge et en effet un `service status jitsi-videobridge` me montre que le bridge s'est lancé mais il avait crash quelques secondes après.
— `/var/log/jitsi/jvb.log` : qui sont les logs du video-bridge. Je vois donc `Error: Could not find or load main class $JVB_EXTRA_JVM_PARAMS` qui se trouve être une variable présente dans `/etc/jitsi/videobridge/config`. J'enlève alors cette variable de ce fichier, je redémarre le bridge... lance un premier appareil, un second, les deux sont présents ! La conférence remarche enfin.

On complète alors la machine avec les outils de monitoring et de back-up, bref, le Jitsi semble bel et bien fonctionnel cette fois.

Pour conclure, si vous voulez héberger un Jitsi :

1. Vous pouvez générer votre certificat Let's Encrypt, puis installer en suivant les instructions d'installation rapide. Cette méthode devrait fonctionner pas pire et l'update se fait de temps à autre avec un apt-get update
2. Si vous voulez automatiser le tout, le rôle de *freedomofpress* marche avec quelques modifications. Pour ma part je compte voir pour fixer la recette avec le problème de démarrage de *nginx* et le problème de configuration, puis d'informer la personne qui a fait le patch de base de mes modifications.

Si des problèmes sur l'instance se font voir, les logs de Jitsi sont pas mal complets et utiles.