MASTANTUONO, Ludo
MEO

CR Suivie Projet

INTRODUCTION

Mon projet étant de mettre en place un **IHM** pour notre outil **Ansible** qui est un outil **d'automatisation** de tâche et **planification** de tâche que nous utilisons déjà. Durant la semaine qui a suivi celle de test, j'ai constaté **plusieurs problèmes**, qu'ils soient **techniques ou environnementaux**. Semaphore, bien qu'étant une IHM intéressante en termes **d'automatisation pour Ansible**, reste limitée face à nos **besoins** fondamentaux. Cette IHM ayant été installée récemment, elle a été testée afin de comprendre et de connaître son fonctionnement.

MISE EN SITUATION DES PROBLÈMES

PROBLÈME TECHNIQUE

Concernant **l'aspect technique** : pourquoi sommes-nous limités ? Globalement, c'est l'IHM qui correspond le mieux à Ansible, mais elle ne gère pas un élément essentiel : **la saisie des variables**.

EXEMPLE

Imaginons que nous voulions installer un package. Nous irions sur **l4y2** (**machine Ansible**) lancer l'installation automatisée d'Ansible, renseigner les variables, etc. (**voir sur l4y2**), afin d'atteindre l'objectif qui est d'installer le package.

Le problème est que **Semaphore** ne peut que **planifier des tâches et les automatiser**. Il ne peut donc pas gérer ce système de saisie de variables, ce qui est **pénalisant**. La seule solution serait de **renseigner à l'avance** les packages infra et de les **modifier** en fonction de ce que l'on installe, ce qui n'est **ni pratique ni efficace**, et génère une perte de temps considérable.

PROBLÈME ENVIRONNEMENTALE

Concernant **l'aspect environnemental** : en réfléchissant à un potentiel schéma pour imaginer la **structure du projet**, nous avons constaté qu'au niveau des **clés SSH**, nous serions fortement gênés. En effet, pour que Semaphore puisse se connecter à **l4y2** (**machine Ansible**), il faudrait également passer par les 40 serveurs de chaque **production** (**p70, p60, p30**).

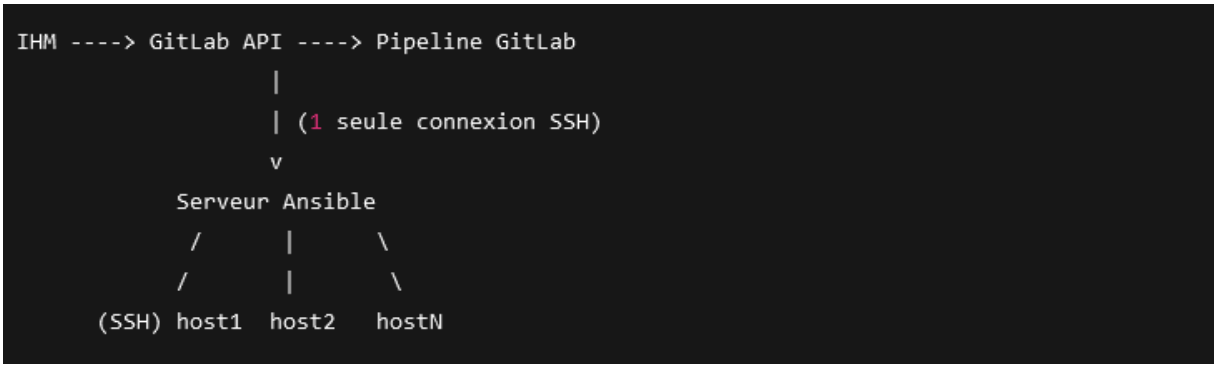
En conclusion, le **CSH** n'accepterait jamais autant de **demandes d'ouverture de ports**, car Semaphore impose un **niveau d'exigence difficilement adaptable** (voire impossible).

SOLUTIONS POSSIBLES POUR RÉALISER LE PROJET

Nous avons réalisé **deux schémas** permettant **d'identifier la structure** du projet et de savoir comment s'y prendre.

1ER SOLUTION

Voici à quoi ressemble ce schéma :



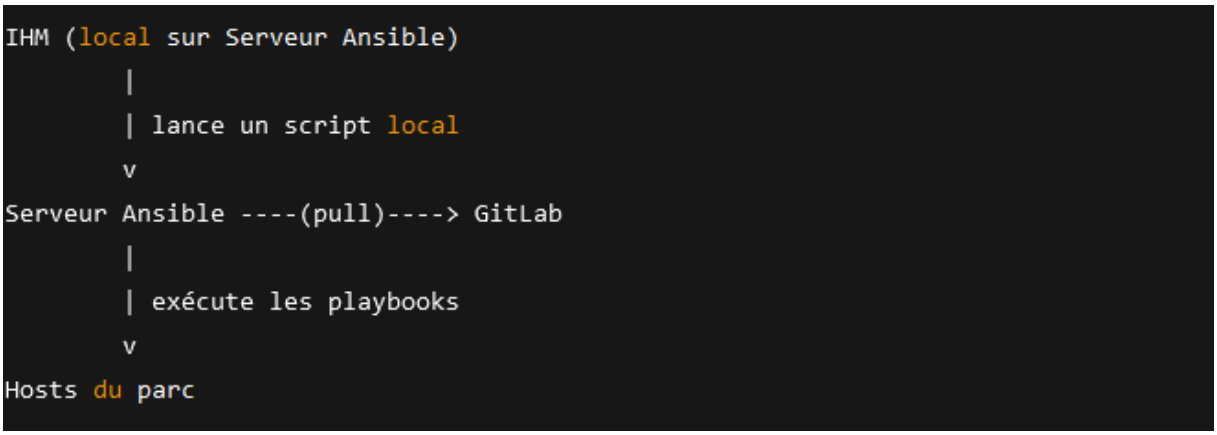
Ce que nous identifions et ce que nous entendons par là, c'est que notre objectif principal est de mettre en place **une seule liaison**, qui reliera le **Git de la MEO**, en demandant une seule ouverture de port au **CSH**. Cela est faisable, et moins il y aura d'ouvertures de ports, mieux ce sera.

La gestion des erreurs côté **pipeline sera réduite**, ce qui permettra **d'identifier plus facilement** l'origine d'un problème.

L'inconvénient que nous rencontrons serait **la dépendance à une unique machine**. De plus, au niveau **des runners**, on pourrait observer de la latence si trop de tâches sont lancées simultanément. (Si quelqu'un lance une tâche pour installer un package et que un autre aussi et encore un ça veut dire que le dernier sera mis en attente)

2EME SOLUTION

Voici à quoi ressemble ce schéma :



Dans ce schéma, plusieurs inconvénients apparaissent. Tout d'abord, en plus du port nécessaire pour Git, il faudrait également **ouvrir un port pour le HTTPS**, puisqu'il faudra gérer des utilisateurs. Cela implique donc davantage de demandes d'ouverture de ports.

Ensuite, le fait que **l'IHM** se trouve **directement** sur la machine Ansible (**l4y2**) pose un problème lorsque nous souhaiterons apporter des **améliorations** à l'IHM ou **corriger** des bugs. En effet, après chaque modification, il faudrait **redemander** une ouverture de port pour pouvoir **déployer la nouvelle version**, car nous serions en **local**. Certes, cela peut sembler pratique, puisqu'on ne passe pas par **l'API Git** ou par **la pipeline**, tout se fait directement sur la machine, mais cela complique considérablement les mises à jour.

CONCLUSION

La solution la plus évidente à retenir serait la **première**, car elle correspond le mieux à nos besoins et présente une **bonne faisabilité**. Pourquoi celle-ci plutôt que la deuxième ? Parce que la seconde solution présente une **faisabilité plus faible**, notamment en raison du **nombre supplémentaire** de demandes d'ouverture de ports, ce qui serait **pénalisant**, voire même impossible à obtenir. Son seul avantage serait d'avoir **l'ensemble du système dans une même "boîte"** (tout en local), mais cet atout ne compense pas **les contraintes** importantes qu'elle impose. Et c'est donc pour ça que je voudrais effectuer une ouverture de port pour que le **GIT** puissent avoir **accès** à **l4y2** et **inversement**.

MASTANTUONO, Ludo
MEO