POKEMON : Perfectly Optimized Kit for Efficient MONitoring

Ce sujet de projet est à réaliser par groupe de 2 ou 3.

La date de rendu est fixée au dimanche 12 janvier 2025 à 23h50. Le rendu des sources, du rapport, et du README se fera via la plateforme Moodle (un rendu par groupe).

Modalités

Dans ce projet (à réaliser en binome de préférence) vous demandera de programmer des plans de contrôle en Python, ainsi qu'un plan de données en P4. Votre projet devra fonctionner sur l'image Docker fournie pour ce module, ou sur la VM de l'ETH Zurich, sans aucune autre dépendance supplémentaire à installer.

Il vous est demandé de rendre votre code accompagné d'un rapport au format PDF. Votre code sera indenté et commenté aux endroits nécessaires. Le rapport expliquera ce que vous avez implémenté et comment, ainsi que ce qui n'a pas été implémenté et pourquoi. Une partie du rapport sera dédiée à la description d'expériences et de tests *reproductibles* qui illustreront les capacités de votre plateforme. Le README détaillera l'architecture du projet ainsi que d'éventuels détails techniques pour lancer votre plateforme.

Le fond comme la forme du rapport (syntaxe, grammaire et orthographe) feront partie intégrante de la notation. La forme du code (indentation, commentaires, lisibilité) sera également prise en compte. Le non-respect des consignes présentées dans cette section entraînera un malus.

Ce projet vous demandera probablement de consulter la documentation P4 ainsi que celle de p4utils. Le dépôt Git P4learning dispose également de ressources qui pourraient vous être utiles.

Description générale

Ce projet a pour objectif, comme son nom l'indique, la conception d'une plateforme de supervision (monitoring) d'un réseau comportant des équipements programmables. Les plans de contrôle standards se concentrant sur des états de liens *on/off*, les défaillances moins prononcées que ces pannes franches, telles que des pertes de paquets non systématiques ou des erreurs de commutation (e.g., dues à des zones mémoire corrompues) peuvent rester invisibles. Cette plateforme doit être en mesure de détecter ces défaillances, ici artificiellement générées à l'échelle d'un routeur.

Bien que chaque équipement possède son propre plan de contrôle, nous nous reposerons également sur l'utilisation d'un contrôleur global, orchestrant les politiques de supervision des commutateurs et chargé de compiler les données métrologiques pour détecter et régair à ces anomalies habituellement sous le radar. Les données seront récoltées via l'utilisation de sondes, émises par les commutateurs et dont les chemins sont prédéfinis par encapsulation. Pour simplifier la réalisation du projet et les tests, nous supposerons que les équipements démarrent défectueux ou sains, et que leur état ne change pas avec le temps.

Vous travaillez sur la topologie de votre choix avec une faible proportion d'équipements défaillants pour simplifier votre implémentation et son évaluation. Veilliez bien à ce que votre évaluation, et donc la topologie sous-jacente, soit la plus générale possible.

Projet

Création des équipements

1) Créez un équipement *simple_router*, capable de faire du routage IP intra-domaine. Cet équipement est composé d'un plan de données P4, mais également d'un plan de contrôle python en charge du calcul des meilleurs chemins et de l'installation des entrées pertinentes dans les tables. On supposera que la topologie est connue de tous les équipements et toujours cohérente.
2) Rajoutez à cet équipement une fonctionnalité d'encapsulation permettant d'imposer des points de passage intermédiaires aux paquets. Ces points de passage seront définis comme des noeuds ou des liens spécifiques à traverser. Concevez une expérience pour vous assurer que les paquets suivent les chemins spécifiés par les points de passage.
3) Créez un équipement *simple_router_loss* et *simple_router_stupid*, qui sont des versions du *simple_router* souffrant d'anomalies; par exemple avec une probabilité de 30% de pertes de paquets, et l'utilisation de chemins aléatoires au lieu des plus courts chemins (afin de simuler un équipement corrumpu). Dans un premier temps, l'équipement *simple_router_stupid* ne fera pas d'erreur si le chemin du paquet est spécifié explicitement lien par lien.

Création du méta-contrôleur

1) Créez un *méta-contrôleur*, ayant connaissance de toute la topologie. Ce méta-contrôleur est en charge d'instancier le contrôleur de chaque équipement du réseau. Le méta-contrôleur doit notamment implémenter les fonctions suivantes :
   a) `read_register_on(switch_id, register_name)`: retourne la valeur du registre `register_name` du commutateur `switch_id`.
   b) `write_register_on(switch_id, register_name, value)`: écrit la valeur `value` dans le registre `register_name` du commutateur `switch_id`.
   c) `install_entry_on(switch_id, table_name, entry)`: installe l'entrée `entry` dans la table `table_name` du commutateur `switch_id`.
   d) `remove_entry_on(switch_id, table_name, entry)`: retire l'entrée `entry` de la table `table_name` du commutateur `switch_id`.

Et tout autre fonction que vous jugerez nécessaire.

Le méta-contrôleur est en charge de communiquer la topologie du réseau aux contrôleurs des commutateurs et est capable de la modifier.

Création d'une politique de supervision basique : analyse du taux de perte

1) Dans un premier temps, chaque commutateur est en charge de superviser tous ses liens sortants. Chaque contrôleur doit envoyer régulièrement un paquet au plan de données (via le port CPU). Le plan de données se chargera de transformer ce paquet en autant de sondes que de liens sortants. Celles-ci auront pour destination le commutateur émetteur mais seront encapsulées *explicitement* (lien par lien) de manière à réaliser un aller-retour sur leur lien sortant.

   Le plan de données de chaque commutateur maintient, pour chaque lien (non-dirigé), différentes informations pertinentes, telles que le nombre de sondes envoyées et le nombre de sondes étant revenues.

2) Régulièrement, le méta-contrôleur demandera aux contrôleurs de remonter les valeurs supervisées qui iront les piocher directement dans leur plan de données. Ces valeurs seront affichées par le méta-controlleur (à la demande) et lorsqu'une anomalie est détectée, cette dernière doit être mise en lumière.

Supervision des chemins

1) On cherche à présent à superviser les plus courts chemins à l'aide de sondes (i.e. leur adéquation avec les meilleures routes supposées). Ces sondes seront également émises par le contrôleur de chaque commutateur, mais seront cette fois diffusées vers l'ensemble des commutateurs du réseau selon des modalités légèrement différentes : aller-retour depuis le commutateur émetteur vers chaque autre commutateur, via une encapsulation non-explicite (relâchée par noeud).

   Ces sondes doivent récolter dans un header dédié l'intégralité des sauts intermédiaires par lesquels elles sont passées. A leurs retours, ces informations sont remontées aux contrôleurs qui les enregistrent dans des structures dédiées.

2) Régulièrement (ou à la demande de l'opérateur), le méta-contrôleur récoltera auprès des contrôleurs les données collectées par les sondes supervisant les chemins. Si certaines sondes ne semblent pas passer par les plus courts chemins, cette anomalie doit être affichée à l'utilisateur.

Détection & Réaction

1) Lorsqu'une anomalie est détectée par le méta-contrôleur, ce-dernier devra maintenant modifier les poids IGP afin que le(s) routeur(s) problématique(s) ne soi(en)t plus utilisé(s). Déterminez l'ensemble de changements minimaux pour atteindre ce nouvel état.

Les deux derniers points

A partir de maintenant, les équipements *simple_router_stupid* peuvent faire des erreurs de commutation sur *tous* les paquets. Vous pouvez également modifier le taux d'équipements défaillants.

1) Optimisez votre plateforme de supervision pour simplifier son fonctionnement, et/ou réduire l'impact de vos mesures (e.g. minimiser la surcharge réseau induite par les sondes) sur le réseau sans dégrader ses capacités de détection (pertes et chemins). Vous détaillerez et justifierez ces améliorations-généralisations dans le rapport.