Réseaux Programmables

Rapport de projet

Perfectly Optimized Kit for Efficient Monitoring (POKEMON)

Brelot Julien

Git

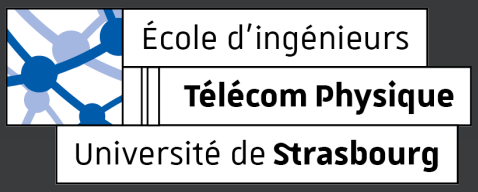

Table des matières

I Introduction

Ce rapport présente le travail réalisé dans le cadre du mini-projet de Réseaux Programmables.
L'objectif de ce projet est de mettre en place un réseau d'equipements réseaux programmables, avec une solution de supervision et de contrôle des liens.
Dans ce but, le data plane est programmé en P4 avec un fichier source unique et générale pour l'ensemble des équipements.
Le control plane est programmé en Python et permet de contrôler les équipements et de collecter les informations de supervision à l'aide d'un méta-contrôleur.
Les routeurs disposent de 3 types controle plane différent : simple, avec perte et une dernier définissant un routage non optimal.

II Réalisations

II.1 Contenu

On retrouve dans ce mini-projet :

- Un fichier P4 pour le data plane des 3 types de routeurs.
- Un méta-contrôleur qui permet de contrôler les routeurs et de collecter les informations de supervision.
- 3 types de controle plane définis en python pour les routeurs : simple$_router, simple_router_lossetsimple_router_stupid.$
- Un fichier python network.py qui permet de générer et lancer un réseau.

II.2 Data Plane

Le **plan de données** est implémenté en P4 et se charge de la gestion des paquets circulant à travers les commutateurs. Voici un résumé de son fonctionnement en relation avec les objectifs du projet POKEMON :

1. Routage intra-domaine

- Chaque commutateur (équipement `simple_router`) effectue un routage IP classique en utilisant des tables pour déterminer les meilleurs chemins.
- Le plan de contrôle installe les entrées nécessaires dans les tables du plan de données pour permettre ce routage.

De ce fait, il n'est pas nécessaire d'avoir un plan de données spécifique pour chaque type de routeur.
Il suffit, pour le plan de contrôle des routeurs simple_router_stupid, de calculer des chemins non nécessairement optimaux et de transmettre les next hops au plan de données.
Pour simuler le comportement de perte de paquets, on aurait pu utiliser la fonction de l'API net de P4 pour simuler des pertes de paquets.
Néanmoins, j'ai choisi d'utiliser une perte de paquets aléatoire via le plan de données afin d'affecter seulement les paquets de données et non les paquets de sondes.
Le plan de contrôle indique simplement au plan de données la taux de perte de paquets via le registre `loss_rate`.

2. Encapsulation pour points de passage intermédiaires

- Le plan de données inclut une fonctionnalité d'encapsulation qui force les paquets à suivre des chemins spécifiques définis par des points de passage (nœuds ou liens).
- Cela est accompli via un en-tête dédié où les points de passage sont empilés dans l'ordre de passage.
- Cette encapsulation impose des points de passages qui peuvent donc être des noeuds non adjacents.
- Les points de passage sont indiqués via leur adresse IP.
- Le paquet est routé vers le prochain point de passage en utilisant le routage standard.
- Une fois que tous les points de passage ont été atteints, le paquet est désencapsulé et routé vers sa destination finale.

3. Gestion des anomalies des équipements

- Deux variantes du `simple_router` sont introduites :
  - `simple_router_loss` : a une probabilité de 30% (configurable) de perte de paquets.
  - Cette perte est modifiable en via le registre `loss_rate` et via le contrôleur simple
  - `simple_router_stupid` : utilise des chemins aléatoires au lieu des meilleurs chemins.
  - Le plan de données est le même pour les 3 types de routeurs, seul le plan de contrôle change.

4. Supervision des liens

- A la réception d'un message de débût de contrôle, le routeur broadcast un message de sondes sur tous ses liens.
- Le message de sonde indique dans l'en-tête, un numéro de protocole IP spécifique pour les sondes de liens.
- A la réception d'une sonde, par un voisin, le voisin renvoie la sonde avec un numéro de protocole différent indiquant un retour par le même lien.
- Le routeur reçoit les sondes de retour, incrémente un registre de sondes reçues et enregistre les ports de réception dans un registre.
- Le plan de contrôle peut alors récupérer ces informations pour déterminer les liens actifs et les pertes de paquets.

5. Supervision des chemins

Sur un principe analogue à la supervision des liens, le plan de contrôle peut envoyer des sondes de chemins pour vérifier la conformité des chemins empruntés par les paquets. Il envoie dans l'en-tête en numéro de protocole ipv4 spécifique pour les sondes de chemins.

- A la réception d'une sonde de supervision de chemin, le routeur encapsule son IP dans une en-tête personnalisée de méta données.
- Il route ensuite le paquet vers le next hop selon la destination du paquet.
- Il modifie finalement le protocole IP du paquet pour indiquer une sonde de supervision aller.
- Chaque point de passage encapsule de même son IP et route le paquet vers le prochain point de passage.
- Une fois atteint la destination, et renvoyé vers l'expéditeur. Et on indique un protocole différent pour le retour.
- Lors d'un retour, les routeurs n'encapasulent pas leur IP et route directement le paquet vers le routeur d'origine.
- À la réception, le routeur renvoie le paquet vers le plan de contrôle qui analysera le paquet et les points de passage pour déterminer la conformité du chemin.

6. Réaction aux anomalies

- En cas d'anomalie (pertes élevées, chemins non conformes), le méta-contrôleur modifie les poids IGP des liens pour contourner les équipements défaillants.
- Ces modifications entrainent un nouveau calcul des chemins par les contrôleurs et une réinstallation des entrées dans les tables du plan de données.
- Les routeurs continuent de fonctionner normalement, mais avec des chemins modifiés.
- De cette façon, on limite l'impact des équipements défaillants sur le réseau et l'interrupion des services.
- Cette réaction incombe donc au méta-contrôleur et aux contrôleurs et reste transparente pour les routeurs.

Points Clefs du Data Plane

Le plan de données agit comme un moteur exécutant les politiques définies par le plan de contrôle. Il se concentre sur :

- La gestion des paquets de routage standard et des sondes.
- La surveillance des liens et des chemins.
- La réaction rapide aux instructions du méta-contrôleur pour garantir un fonctionnement réseau optimal, même en présence d'équipements défaillants.

II.3 Control Plane

Méta-contrôleur

1.Rôle du méta-contrôleur

Le méta-contrôleur coordonne les actions des contrôleurs locaux associés à chaque switch du réseau, gère la supervision et l'optimisation des liens du réseau. Il intervient principalement pour :

- Superviser les liens actifs du réseau.
- Diagnostiquer et réagir aux anomalies de routage et de pertes de paquets.
- Envoyer des sondes pour surveiller l'état des chemins.

2. Rôle du méta-contrôleur

Le méta-contrôleur coordonne les actions des contrôleurs locaux associés à chaque switch du réseau, gère la supervision et l'optimisation des liens du réseau. Il intervient principalement pour :

- Superviser les liens actifs du réseau.
- Diagnostiquer et réagir aux anomalies de routage et de pertes de paquets.
- Envoyer des sondes pour surveiller l'état des chemins.

3. Fonctionnement principal

Le méta-contrôleur effectue les tâches suivantes :

a) Supervision périodique

- **Réinitialisation des registres** : Effacement des données collectées précédemment.
- **Envoi de sondes** :
  - Les contrôleurs locaux envoient des paquets de sonde pour mesurer la qualité des chemins.
  - Les ports CPU des switches capturent et analysent ces paquets.
- **Sniffing des paquets** : Analyse des paquets reçus pour collecter des statistiques et détecter des anomalies.

b) Collecte et analyse des anomalies

- **Statistiques collectées** :
  - Taux de pertes par port.
  - Chemins empruntés versus chemins optimaux.
- **Détection d'anomalies** :
  - Ports ou liens inactifs.
  - Utilisation de chemins sous-optimaux.

c) Réactions aux anomalies

- **Suppression de liens défectueux** : Les liens associés aux ports inactifs sont désactivés.
- **Réajustement des poids** : Augmentation des poids des chemins sous-optimaux pour pénaliser leur utilisation.

4. Gestion des registres et des tables

Le méta-contrôleur offre des méthodes pour :

- Lire ou écrire dans les registres des switches.
- Ajouter ou supprimer des entrées dans les tables de routage.

5. Diagnostic des anomalies

Les contrôleurs locaux disposent de méthodes pour :

- Identifier les liens défectueux.
- Comparer les chemins optimaux et réels.
- Générer des alertes ou ajuster la topologie.

6. Architecture modulaire

Le méta-contrôleur centralise la supervision tout en déléguant les tâches aux contrôleurs locaux pour assurer la scalabilité.

Contrôleurs

Le contrôleur standard assure le fonctionnement d'un routeur simple, capable de gérer les tâches essentielles comme le routage des paquets, la supervision des liens et la détection des anomalies réseau. Il repose sur un **plan de données P4** et un **plan de contrôle Python**.

1. Initialisation

- **Compilation du fichier P4** : Compile les définitions des tables et actions pour le plan de données.
- **Chargement de la topologie réseau** : Initialise un graphe réseau basé sur un fichier JSON ou une structure `NetworkGraph`.
- **Configuration des registres** : Les registres comme `loss_rate`, `total_packets_lost`, etc., sont initialisés à zéro.
- **Déploiement initial** :
  - Calcul des routes.
  - Installation des règles de routage (`ipv4_lpm`) et d'informations du routeur (`router_info`).
  - Configuration des règles de multicast pour les sondes.

2. Routage

- **Calcul des chemins optimaux** : Utilise les algorithmes de plus court chemin pour déterminer les routes entre les nœuds du réseau.
- **Installation des règles** :
  - Les règles associent une adresse IP de destination, un port de sortie, une adresse MAC et éventuellement une adresse IP source.
  - Les hôtes reçoivent des règles spécifiques en fonction de leur IP.
- **Mise à jour des tables** : Les règles obsolètes ou incorrectes peuvent être supprimées ou réinstallées.

3. Supervision

- **Envoi de sondes** :
  - Les sondes de test des liens (`PROTOCOL_LINK_TEST_TRIGGER`) sont diffusées sur tous les ports.
  - Les sondes de test des chemins (`PROTOCOL_PATH_TEST_TRIGGER`) sont envoyées vers des destinations spécifiques.
- **Capture des réponses des sondes** :
  - Le contrôleur écoute les paquets reçus sur le port CPU.
  - Les informations des chemins empruntés sont extraites des paquets et comparées aux chemins optimaux.

4. Diagnostic

- **Anomalies sur les liens** :
  - Vérifie les ports fonctionnels en comparant les registres `links_up` avec la configuration attendue.
  - Identifie les ports inactifs (liens down).
- **Anomalies sur les chemins** :
  - Compare les chemins optimaux avec ceux réellement empruntés par les sondes.
  - Rapporte les écarts, notamment les chemins non optimaux.
- **Rapports statistiques** :
  - Génère des rapports détaillés incluant les ports inactifs, les chemins incorrects, le taux de perte de paquets, et les statistiques de sondes.

5. Réinitialisation

- **Réinitialisation complète** : Les tables et registres peuvent être nettoyés et réinitialisés en cas de redémarrage ou de mise à jour de la topologie.

Contrôleurs spécifiques

Les contrôleurs `simple_router_loss` et `simple_router_stupid` sont des variantes du contrôleur standard qui ajoutent des fonctionnalités de perte de paquets et de routage non optimal.
Le contrôleur `simple_router_loss` simule une perte de paquets aléatoire en modifiant simplement le taux de perte dans le registre `loss_rate`.
Le contrôleur `simple_router_stupid` utilise des chemins aléatoires au lieu des chemins optimaux pour simuler un comportement non optimal. Il modifie donc simplement la méthode de calcul des chemins optimaux pour choisir des chemins aléatoires.
Ainsi, la plupart des fonctionnalités du contrôleur standard sont conservées, seul le calcul des chemins ou le taux de perte est modifié.

Points clés du contrôleur

- **Contrôle basé sur des règles dynamiques** : Les tables `ipv4_lpm` et `router_info` sont utilisées pour définir les comportements de routage et la gestion des informations des routeurs.
- **Récupération et mise à jour des registres** : Les registres stockent des données essentielles comme le taux de perte ou l'état des liens, permettant un diagnostic précis.
- **Gestion parallèle** : Des threads distincts assurent la capture des paquets sur le port CPU et la supervision en temps réel.

Points clés du contrôle plane

Comme demandé, le **plan de contrôle** est implémenté en Python et se charge de la gestion des équipements et de la collecte des informations de supervision.
ela passe d'abord par un méta-contrôleur qui gère l'ensemble du réseau et des contrôleurs qui gèrent les routeurs individuellement.
e méta contrôleur initie chaque contrôleur selon la topologie du réseau, initie à une fréquence régulière, une vérification des liens puis des chemins.
l peut aussi réagir aux anomalies en modifiant les poids IGP des liens pour contourner les équipements défaillants.
es contrôleurs, quant à eux, gèrent les routeurs individuellement en leur envoyant des instructions pour la collecte des informations de supervision.
ls analysent ensuite ces informations pour déterminer les liens actifs, les pertes de paquets et la conformité des chemins et remonte cela au méta-contrôleur.
e méta contrôleur peut alors réagir en conséquence pour garantir un fonctionnement optimal du réseau. e dernier transmet alors la nouvelle topologie aux contrôleurs qui réinstallent les entrées dans les tables du plan de données après la modification des poids IGP.

II.4 Déploiement du projet

Le projet peut être cloné depuis le dépôt git suivant :

```
git clone https://github.com/BJCode-git/Projet-Reseaux-Programmables -b mai
cd Projet-Reseaux-Programmables
```

Lancement du conteneur P4 :

```
docker compose run pokemon
```

Démarrage du réseau et du méta-contrôleur / contrôleurs :

```
./start.sh
```