ZSUT

Zakład Sieci i Usług Teleinformatycznych

Podstawy komutacji cyfrowej

Instrukcja
Ćwiczenie: Sterowanie siecią

ZSUT. Zakład Sieci i Usług Teleinformatycznych
Instytut Telekomunikacji
Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska, Grudzień 2020

Cel ćwiczenia

Celem ćwiczenia jest wykorzystanie wiedzy na temat klientów REST do sterowania siecią.

Wstęp

Niestety opanowany przez naszego bohatera mechanizm kolejek nie wystarczył, gdyż szef po negocjacjach z Japończykami popadł w paranoję i zażądał wyłączności na wykorzystywane przez niego do priorytetowych celów łącza. W praktyce oznaczało to, że po mailu o treści "Jutro w godzinach 14-15 mamy Holendrów" należało upewnić się, że ruch z sali konferencyjnej nie będzie się mieszał z innym ruchem. Nie byłby to wielki problem, gdyby większość maili nie wyglądała trochę inaczej. "Gadam z Chińczykami, daj mi priorytet" czy "jestem u Staszka, daj nas na priorytet" okazało się normą i "na priorytet" trzeba było dawać szybko, często i w różnych miejscach. Nasz zaradny bohater przypomniał sobie swoje triumfy na europejskich parkietach giełdowych i zdecydował się proces "dawania na priorytet" zautomatyzować, jak kiedyś handel udziałami.

Jak w przypadku kolejek, procedurę wpierw przetestował na Mininecie. Po pierwszych próbach zorientował się, że sterowanie siecią za pomocą komend wysyłanych bezpośrednio do przełączników jest uciążliwe. Z drugiej strony, zdawał sobie sprawę, że przecież nie jest jedynym administratorem na świecie i pewnie nie on pierwszy stoi przed podobnym problemem. Zdecydował się na otwarty sterownik ONOS, z którym komunikował się za pomocą REST API.

Jedną z funkcjonalności ONOSa, którą administrator zdecydował się użyć, był mechanizm intencji. Intencja w sterowniku ONOS rozumiana jest jako chęć przesyłania pakietów pomiędzy parą adresów. ONOS automatycznie wyznacza ścieżki dla każdej zaobserwowanej intencji. Do intencji możemy przypisać przepustowość, dzięki czemu sterownik będzie mógł lepiej rozplanowywać zapotrzebowania w sieci. Z założenia intencje powinny być generowane przed wygenerowaniem ruchu, ale ONOS może też działać w trybie reaktywnym sam tworząc intencje dla zaobserwowanych pakietów. Z tego drugiego trybu skorzystał nasz administrator, a problem "dawania na priorytet" rozwiązał poprzez wyznaczanie nowych ścieżek dla intencji w taki sposób, że intencja, z której korzysta szef, jest intencją priorytetową i ścieżka dla niej wyznaczana jest jako pierwsza, a pozostałe intencje nie mogą korzystać z przypisanych do niej łączy. W kolejnym kroku administrator za pomocą ONOS API wymusił na sterowniku przekierowanie intencji na wyliczone ścieżki.

Treść zadania

W czasie laboratorium wcielimy się w rolę naszego administratora. Celem ćwiczenia jest napisanie klienta REST, który w przemyślany sposób będzie sterował siecią. Kod klienta należy umieścić na wydziałowym gitlabie:

https://gitlab-stud.elka.pw.edu.pl

Projekt należy zatytułować **2020Z_PKC_ONOS**. Proszę nie zapomnieć o dodaniu do projektu prowadzącego laboratorium, tj. **mzotkiew.** Laboratorium składa się z pięciu kroków. Trzy ostatnie powinny zakończyć się commitami.

Dodatkowo proszę przez System Rezerwacji ZTiT wysłać sprawozdanie, które składać się ma z następujących elementów:

A) zrzut ekranu z wyliczonymi najkrótszymi ścieżkami

B) zrzut ekranu z wyliczoną najkrótszą ścieżką dla intencji priorytetowej oraz z wyliczonymi ścieżkami dla pozostałych intencji.

C) zrzut ekranu ze ścieżką jednej z intencji niepriorytetowych, która nie jest najkrótszą możliwą ścieżką w ogólności; ścieżka powinna być wyświetlona w interfejsie graficznym ONOSa.

D) (opcjonalnie) czas spędzony na poszczególnych częściach tego ćwiczenia oraz ćwiczeń 3a, 3b i 3c; komentarze do treści ćwiczeń.

Przebieg ćwiczenia

Ćwiczenie można wykonywać samodzielnie lub w grupach dwuosobowych. Ćwiczenie wykonujemy na platformie dostępnej przez System Rezerwacji ZTiT.

1. Przygotuj środowisko

1.1 Uruchom ONOSa

Wpisz poniższą komendę w terminalu. Nie zamykaj terminala.

```
sudo /opt/onos/bin/onos-service start
```

Komenda uruchomi sterownik sieci ONOS. Uruchomienie sterownika nie jest natychmiastowe i trwa około minuty. Interfejs graficzny sterownika dostępny będzie po uruchomieniu pod adresem:

http://127.0.0.1:8181/onos/ui

Login: onos, hasło: rocks

1.2 Uruchom skrypt Minineta

Otwórz nowy terminal. Wpisz poniższą komendę. Nie zamykaj terminala.

```
sudo python onos_lab.py
```

Skrypt stworzy w Mininecie sieć, na której będziemy dalej pracować. Sieć powinna pojawić się w interfejsie graficznym ONOSa. Na tym etapie ONOS jeszcze nie wie o istnieniu hostów, więc nie może ich wyświetlić.

1.3 Zasygnalizuj intencje

W terminalu, na którym uruchomiony został Mininet, wpisz poniższą komendę.

```
mininet> pingall
```

Komenda spowoduje wysłanie pakietów ping pomiędzy wszystkimi parami hostów w sieci. Nowe pakiety zostaną zinterpretowane przez ONOS jako nowe intencje, które zostaną skierowane na domyślne ścieżki. Nowe intencje, jak i przypisane do nich ścieżki, można obejrzeć korzystając z interfejsu graficznego ONOSa. Na tym etapie hosty powinny być już widoczne w interfejsie graficznym. Włączyć i wyłączyć wyświetlanie hostów można za pomocą klawisza 'H' na klawiaturze.

W kolejnym kroku musimy zarejestrować intencje w module monitorowania. W tym celu należy otworzyć konsolę zarządzania ONOSa.

```
ssh -p 8101 karaf@127.0.0.1 (hasło: karaf)
```

W konsoli należy wpisać. Podany poniżej numer `184` nie zawsze jest prawidłowy. Problem można rozwiązać dwukrotnie wciskając przycisk `<Tab>` po wpisaniu `imr:startmon`

```
karaf@root> imr:startmon 184 org.onosproject.ifwd
```

Od tego momentu można monitorować i sterować intencjami z wykorzystaniem ONOS API.

UWAGA: ONOS jest oprogramowaniem eksperymentalnym i nie zawsze będzie działał, jak sobie tego życzymy. Problemem mogą być na przykład usunięte intencje, które ciągle zostają zwracane przez moduł monitorujący. Najprostszą drogą rozwiązania tego problemu jest wyłączenie i ponowne włączenie modułu monitorującego (Applications->Intent Monitoring and Rerouting). Po restarcie należy ponownie zarejestrować intencje w module.

2. Ściągnij dane o sieci

Napisz program, który użyje API ONOSa do ściągnięcia informacji o sieci (przełączniki, hosty, łącza, intencje) i stworzy model sieci w pamięci. Uproszczona dokumentacja API ONOSa znajduje się pod poniższym adresem.

https://documenter.getpostman.com/view/12940619/TVspkVLK

Pełna dokumentacja API znajduje się w różnych miejscach na stronie ONOSa [1]. Funkcjonalność udostępniającą informacje o sieci można przetestować korzystając z przeglądarki albo zainstalowanego na maszynie Postmana.

Do reprezentacji sieci, a później do wyliczania ścieżek, można użyć kody ***shortest.py*** znajdującego się na pulpicie maszyny.

3. Wyznacz najkrótszą ścieżkę

Zmodyfikuj program, aby wyznaczał najkrótsze ścieżki dla wszystkich intencji; wypisz te ścieżki na ekran (commit z kodem, zrzut ekranu do sprawozdania). Najkrótszą ścieżkę można wyznaczyć korzystając z kodu ***shortest.py*** znajdującego się na pulpicie maszyny albo samodzielnie implementując algorytm najkrótszej ścieżki np. algorytm Dijkstry [3].

4. Wyznacz pozostałe ścieżki

Zmodyfikuj program, aby wyznaczał najkrótszą ścieżkę dla intencji priorytetowej reprezentowanej przez parę adresów IP (dane wejściowe do programu), a dla pozostałych wyznaczał najkrótsze ścieżki, które nie kolidują z wyznaczoną ścieżką intencji priorytetowej (commit z kodem, zrzut ekranu do sprawozdania).

UWAGA: realizując ten punkt nie powinniśmy modyfikować algorytmu najkrótszej ścieżki, tylko odpowiednio zmienić dostarczane do algorytmu dane wejściowe.

5. Wysteruj ścieżki

Zmodyfikuj program, aby użył API ONOSa i wysterował intencje na wyliczone w poprzednim punkcie ścieżki. (commit z kodem, do sprawozdania zrzut ekranu z interfejsem graficznym ONOSa pokazujący wyliczoną ścieżkę dla jednej z niepriorytetowych intencji, która nie jest ścieżką najkrótszą w ogólności).

Punktacja

33% - napisanie klienta REST, który ściągnie dane o sieci i wyznaczy najkrótsze ścieżki (commit)

33% - napisanie programu, który wyznaczy ścieżki biorąc pod uwagę intencję priorytetową (commit)

33% - napisanie klienta REST, który wysteruje ścieżki (commit)

Czas realizacji: 10.01.2021 23:59

Literatura

[1] https://opennetworking.org/onos/

[2] https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm