Sieci Sterowane Programowo

Adam Twardosz, Dawid Dzhafarov, Mateusz Zięba, Maciej Kornaus, Wiktoria Martyńska

25.01.2024

Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
AGH University of Krakow

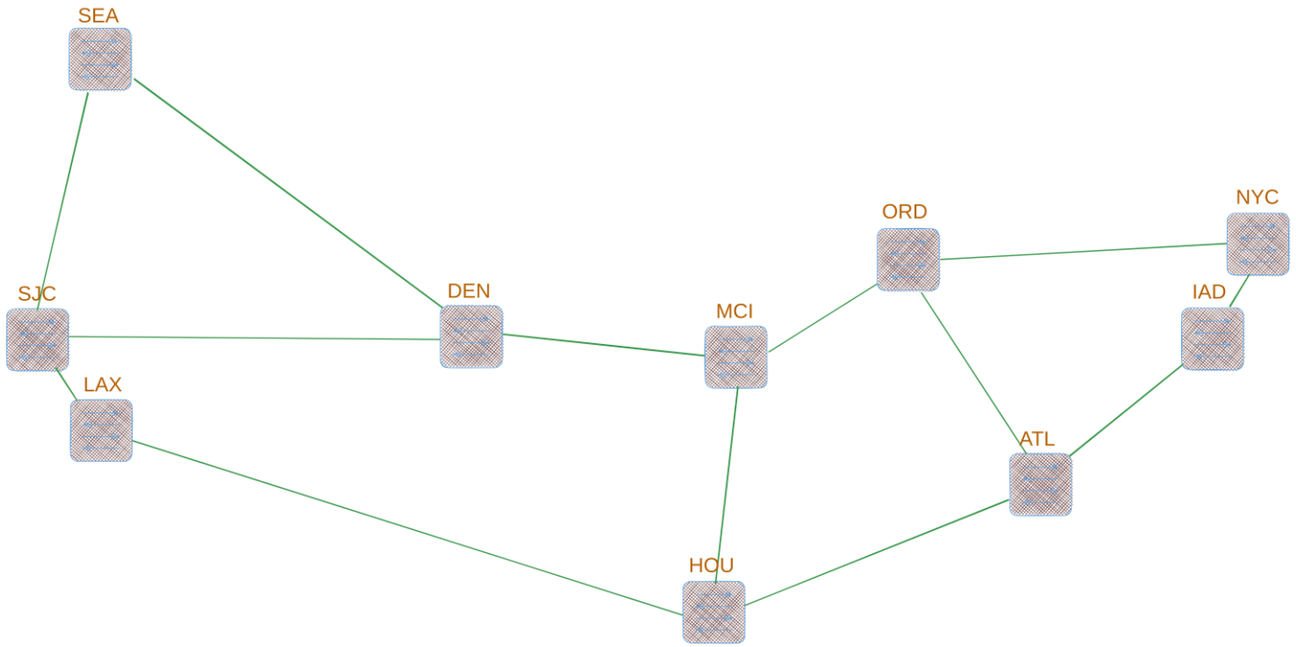
Cel projektu

Hierarchical SDN controllers

- Implementacja hierarchicznej architektury kontrolerów i przełączników
- Przetestowanie działania load-balancera zarządzającego ruchem w sieci w sytuacji wzmożonego ruchu w danym przełączniku
- Zaobserwowanie zmiany w przełączaniu ruchu

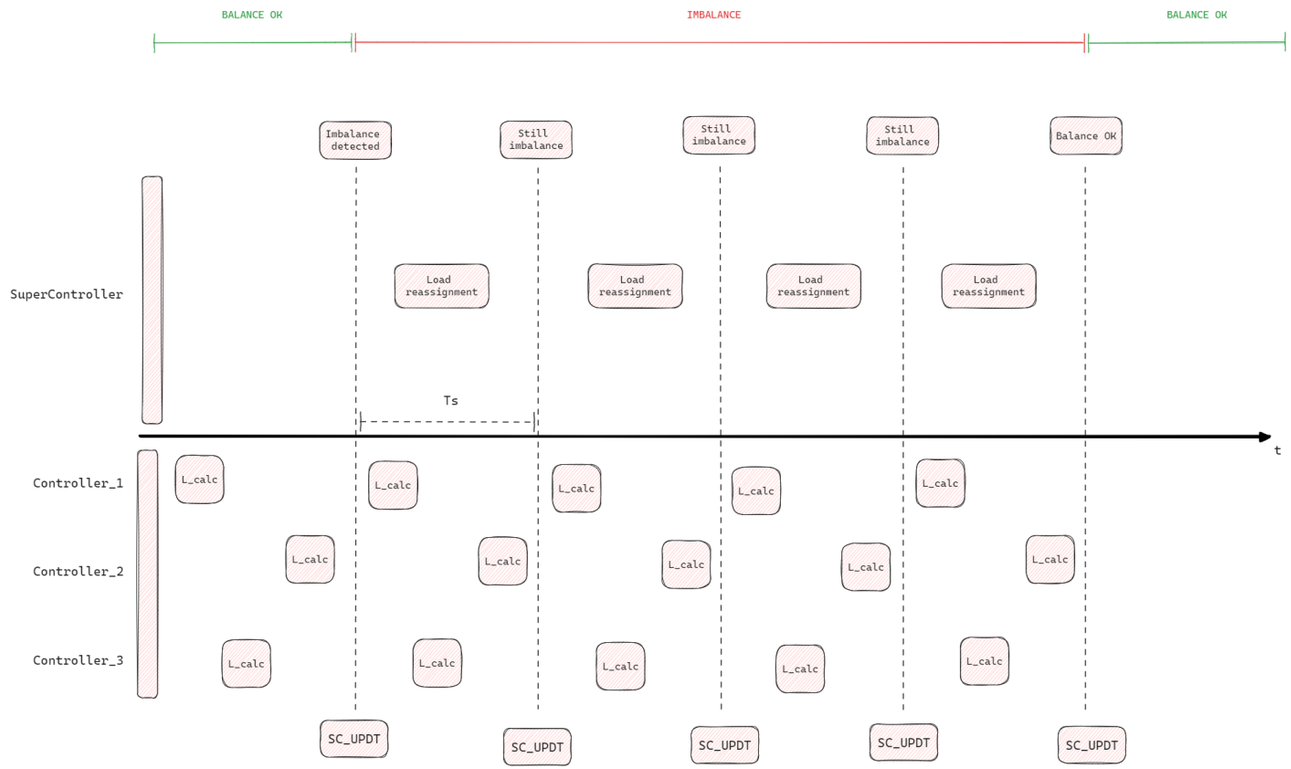
Topologia

- 10 przełączników
- 10 hostów
- 3 kontrolery
- 1 super-kontroler



Algorytm

- Wysyłanie okresowych stanów ramu oraz cpu
- Wykrywanie "braku balansu" poprzez sprawdzanie obciążenia
- Przenoszenie przełączników z bardziej obciążonych na mniej obciążone sterowniki



Generowanie ruchu

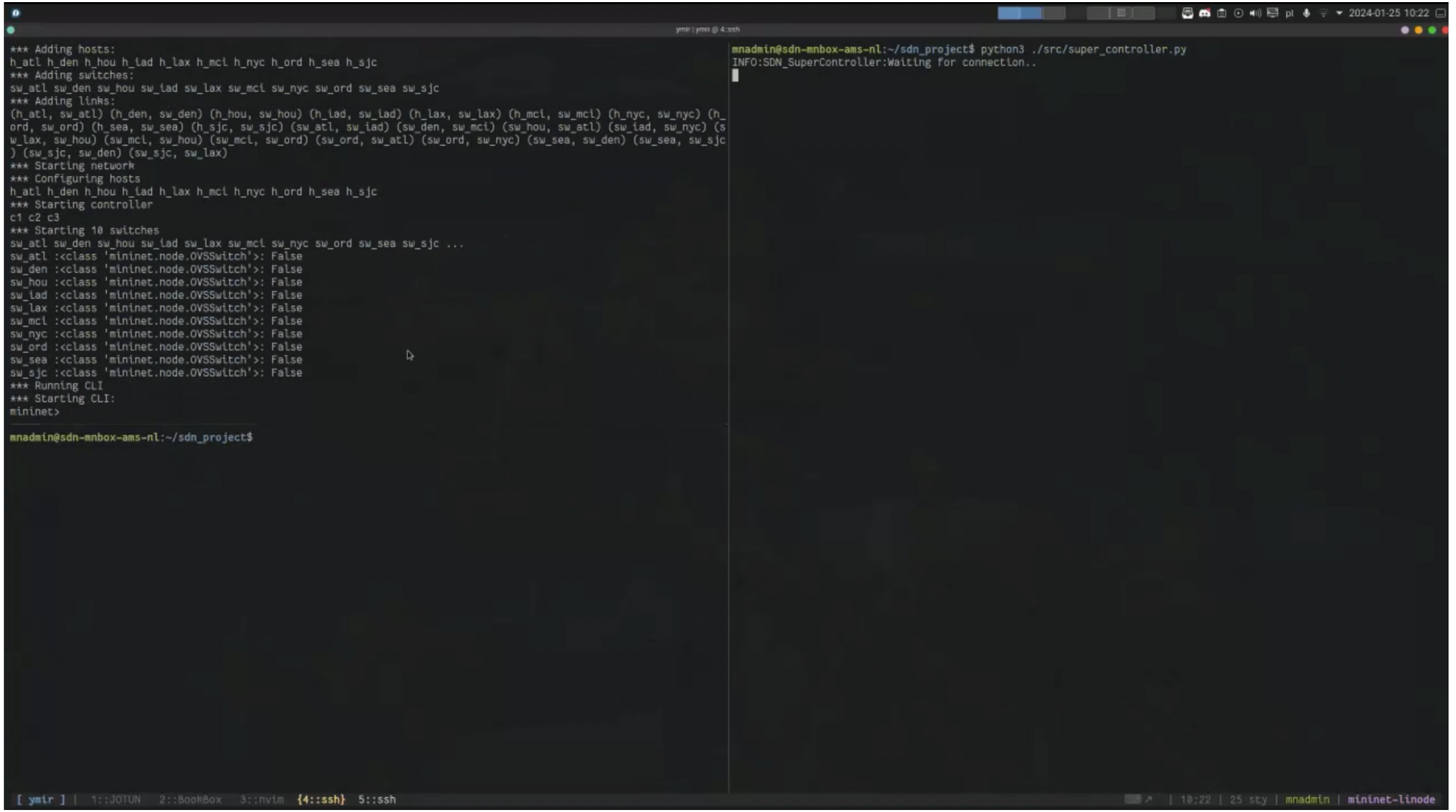
- Użyty generator ruchu: D-ITG-2.8.1-r1023 (https://traffic.comics.unina.it/software/ITG/)
- Generacja 3600 flowów z hosta przyłączonego do przełącznika SJC, do hosta docelowego przyłączonego do przełącznika MCI
- Inne, dobrane parametry ruchu:
  - Czas transmisji: 10s
  - Rozmiar pakietu: 10B
  - Ilość pakietów na sekundę: 1-100

Działanie

- Przy wykryciu obciążenia, superkontroler wysyła do danego kontrolera żądanie o zmianę roli na SLAVE w pierwszym z podległych mu przełączników
- Proces jest powtarzany, dopóki zniknie problem obciążenia

```python
def _balanceControllers(self, busy_worker_id: int, free_worker_id: int) -> None:
    """Move one switch from busy to free.

    Args:
        busy_worker_id: id of a busy controller/worker
        free_worker_id: id of a free controller/worker
    """
    for dpid, role in self.workers[busy_worker_id].dpid2role.items():
        if role [ROLE.MASTER.value, ROLE.SLAVE.value] and dpid in self.workers[free_worker_id].dpid2role.keys():
            msg = json.dumps({
                'cmd': f"{CMD.ROLE_CHANGE}",
                'role': ROLE.SLAVE.value,
                'dpid': dpid,
            })
            self.workers[busy_worker_id].sendMsg(msg)
            self.workers[busy_worker_id].dpid2role[dpid] = ROLE.SLAVE
            msg = json.dumps({
                'cmd': f"{CMD.ROLE_CHANGE}",
                'role': ROLE.MASTER.value,
                'dpid': dpid,
            })
            self.workers[free_worker_id].sendMsg(msg)
            self.workers[free_worker_id].dpid2role[dpid] = ROLE.MASTER
```


2024-01-25 10:22
ymir | ymir @ 4::ssh
*** Adding hosts:
h_atl h_den h_hou h_iad h_lax h_mci h_nyc h_ord h_sea h_sjc
*** Adding switches:
sw_atl sw_den sw_hou sw_iad sw_lax sw_mci sw_nyc sw_ord sw_sea sw_sjc
*** Adding links:
(h_atl, sw_atl) (h_den, sw_den) (h_hou, sw_hou) (h_iad, sw_iad) (h_lax, sw_lax) (h_mci, sw_mci) (h_nyc, sw_nyc) (h_ord, sw_ord) (h_sea, sw_sea) (h_sjc, sw_sjc) (sw_atl, sw_iad) (sw_den, sw_mci) (sw_hou, sw_atl) (sw_iad, sw_nyc) (sw_lax, sw_hou) (sw_mci, sw_hou) (sw_mci, sw_ord) (sw_ord, sw_atl) (sw_ord, sw_nyc) (sw_sea, sw_den) (sw_sea, sw_sjc) (sw_sjc, sw_den) (sw_sjc, sw_lax)
*** Starting network
*** Configuring hosts
h_atl h_den h_hou h_iad h_lax h_mci h_nyc h_ord h_sea h_sjc
*** Starting controller
c1 c2 c3
*** Starting 10 switches
sw_atl sw_den sw_hou sw_iad sw_lax sw_mci sw_nyc sw_ord sw_sea sw_sjc ...
sw_atl :<class 'mininet.node.OVSSwitch'>: False
sw_den :<class 'mininet.node.OVSSwitch'>: False
sw_hou :<class 'mininet.node.OVSSwitch'>: False
sw_iad :<class 'mininet.node.OVSSwitch'>: False
sw_lax :<class 'mininet.node.OVSSwitch'>: False
sw_mci :<class 'mininet.node.OVSSwitch'>: False
sw_nyc :<class 'mininet.node.OVSSwitch'>: False
sw_ord :<class 'mininet.node.OVSSwitch'>: False
sw_sea :<class 'mininet.node.OVSSwitch'>: False
sw_sjc :<class 'mininet.node.OVSSwitch'>: False
*** Running CLI
*** Starting CLI:
mininet>
mnadmin@sdn-mnbox-ams-nl:~/sdn_project$
mnadmin@sdn-mnbox-ams-nl:~/sdn_project$ python3 ./src/super_controller.py
INFO:SDN_SuperController:Waiting for connection..
[ymir] | 1::JOTUN 2::BookBox 3::nvim {4::ssh} 5::ssh
| 10:22 | 25 sty | mnadmin | mininet-linode

Napotkane problemy

- Konieczność głębokiej ingerencji w działanie Ryu
- Problemy z topologią i z uruchomieniem trzech kontrolerów jednocześnie
- Problem z wymuszeniem konkretnej roli dla kontrolera
- Ryu nie jest już wspierane, więc napotkaliśmy problemy z instalacją: w celu zainstalowania sterownika na maszynej wirtualnej musieliśmy przerobić kod źródłowy
- Uruchomienie generatora ruchu także wymagało ingerencji w kod źródłowy

Bibliografia

• Cooperative Load Balancing for Hierarchical SDN Controllers, Hakan Selvi, G¨urkan G¨ur, and Fatih Alag¨oz, data dostępu: 26.10.2023r

- D-ITG: https://traffic.comics.unina.it/software/ITG/
- ryu: https://github.com/faucetsdn/ryu