Wojskowa Akademia Techniczna

Wydział Elektroniki

**Bezpieczeństwo informacyjne w sieciach programowalnych**

**(Data-Centric Security in Software Defined Networks)**

Opis środowiska

Opracował:

dr inż. Konrad Wrona

mgr inż. Sebastian Szwaczyk

Warszawa 2017

**1. Wstęp**

Laboratoria w całości zostaną przeprowadzone wykorzystując komputery klasy PC z zainstalowanym systemem operacyjnym DEBIAN 9. Jako główną platformę do wykonania ćwiczenia wykorzystano technologię Docker.

**2. Instalacja środowiska**

Dzięki technologii Docker każdy ze studentów w prosty sposób może uruchomić środowisko testowe na swoim prywatnym komputerze.

Wymagania:

* Komputer klasy PC z systemem operacyjnym z rodziny Debian/Ubuntu (lub inny Linux z modułem openvswitch standardowo dołączonym do jądra) lub Mac OS
* Zainstalowany Docker CE (<https://www.docker.com/>)
* Zainstalowany klient GIT (<https://pl.wikipedia.org/wiki/Git_(oprogramowanie>))

Do przeprowadzenia zajęć laboratoryjnych został przygotowany obraz Dockera zawierający niezbędne oprogoramowanie. Obraz Dockera dostępny jest w repozytorium (<https://hub.docker.com/r/sszwaczyk/docker-sdn-labs/>), aby go pobrać należy wkonać komendę:

*docker pull sszwaczyk/docker-sdn-labs*

Po pobraniu obraz jest gotowy do użycia. W celu ułatwienia korzystania z tego obrazu przygotowano skypty pozwalające na uruchomienia, dołączenia oraz zatrzymanie kontenera. Skrypty dostępnę są poprzez repozytorium github (<https://github.com/sszwaczyk/docker-sdn-labs>), aby je pobrać należy wykonać komendę:

*git clone https://github.com/sszwaczyk/docker-sdn-labs.git*

Zostanie pobrany folder ze skryptami (run.sh, attach.sh, stop.sh).

**3. Praca z kontenerem**

Na komputerach w laboratorium wszystkie skrypty wykorzystywane w instrukcji znajdują się w katalogu */home/student/SDN/docker-sdn-labs.*

**3.1 Uruchamianie**

Aby uruchomić kontener wystarczy uruchomić skrypt run.sh. Zawartość tego skrpytu:

*#!/bin/bash*

*xhost +*

*docker run -tid --privileged --rm -e DISPLAY=$DISPLAY -v /tmp/.X11-unix:/tmp/.X11-unix --name=sdn-labs sszwaczyk/docker-sdn-labs*

*attach.sh*

Skrypt wykonuje polecenie run w celu uruchomienia wcześniej pobranego obrazu sszwaczyk/docker-sdn-labs. Następnie łączy się do uruchomionego kontenera. Opis pozostałych opcji można znaleźć w pomocy. Po uruchomieniu kontenera w konsoli powinień pojawić się jego ID:

*$ ./run.sh*

*access control disabled, clients can connect from any host*

*c74500f93de6b63e25ae556afbc9672500ac92d464be4a769bc4da4eafbaf31f*

*To run a command as administrator (user "root"), use "sudo <command>".*

*See "man sudo\_root" for details.*

[*student@c74500f93de6*](mailto:student@c74500f93de6)*:/$*

Uruchomione kontenery można sprawdzić przy pomocy polecenia:

*docker ps*

Przykładowy wynik komendy docker ps:

*$ docker ps*

*CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES*

*9f22d9128c03 sszwaczyk/docker-sdn-labs "/bin/sh -c 'servi..." About a minute ago Up About a minute sdn-labs*

**3.2 Łączenie z kontenerem**

W celu połączenia z kontenerem wystarczy użyć skryptu attach.sh. Zawartość skryptu:

*docker exec -t -i prc-labs bash -c 'su student'*

Po połączeniu dostajemy dostęp do bash jako użytkownik student.

**3.3 Łączenie z kontenerem po ssh**

W celu połączenia z kontenerem wystarczy użyć skryptu attach.sh. Zawartość skryptu:

*#!/bin/bash*

*ssh -X* [*student@172.17.0.2*](mailto:student@172.17.0.2)

Jak widać skrypt próbuje nawiązać połączenie ssh jako użytkownik student z adresem ip 172.17.0.2. Jest to domyślny adres ip w przypadku gdy uruchomiony jest tylko jeden kontener. Wykorzystwana jest tu również opcja -X pozwalająca na przesyłanie obrazu dla aplikacji graficznych (w przypadku wystąpienia problemów z przekazywaniem obrazu dla systemu Mac OS, można spróbować <https://fredrikaverpil.github.io/2016/07/31/docker-for-mac-and-gui-applications/>). Po nawiązaniu połączenia, należy podać hasło dla użytkownika student. Hasło to **student.**

Po udanym zalogowaniu można sprawdzić poprawność działania przesyłania obrazu wydając polecenie *wireshark*. Jeśli wszystko działa poprawnie powinno pojawić się GUI sniffera wireshark.

Połączenie można zakończyć poleceniem *exit*. Rozłączenie połączenia ssh nie jest jednoznaczne z wyłaczeniem kontenera!

**3.4 Zatrzymanie kontenera**

W celu zatrzymania działającego kontenera należy użyć skryptu stop.sh. Skrypt ten wykonuje jedną komendę:

*#!/bin/bash*

*docker stop sdn-labs*

Zatrzymuje on kontener o nazwie sdn-labs (ta sama nazwa została użyta przy poleceniu run oraz widać ją w wyniku wykonania komendy docker ps).

UWAGA!!! - Po zatrzymaniu kontenera wszystkie pliki na nim zapisane są tracone!!! Jeśli podczas pisania instukcji wyjdzie, że studenci będą musieli zapisać jakieś pliki to zrobi się to poprzez zamontowanie jednego z katalogów hosta do kontenera.

**3.4 Uruchamianie programów bez łączenia**

Przygotowano również skrpyty pozwalające na uruchomienie eclipse i wireshark bezpośrednio:

wireshark\_root.sh

wireshark\_student.sh

eclipse\_root.sh

eclipse\_student.sh

Wystarczy wywołać odpowiedni ze skryptów i program zostanie uruchomiony z odpowiednimi uprawnieniami.

**4. Struktura obrazu**

W ramach zajęć będzie wykorzystywany następujący stos technologiczny:

* mininet → narzędzie do emulacji sieci SDN
* OpenVSwitch → implementacja przełącznika SDN
* Floodlight → implementacja sterownika SDN
* Eclipse → środowisko programistyczne

**4.1 Mininet**

Mininet jest narzędziem pozwającym na emulacje sieci SDN. Sieć tworzona jest na poziomie jądra systemu operacyjnego. W celu stworzenia sieci mininet posługuje się interfejsami wirtualnymi oraz wydzielanymi przestrzeniami nazw. Aby sprawidzć poprawność działania minineta wydaj polecenie:

*sudo mn*

po chwili powinna pojawić się informacja o uruchomieniu domyślnej topologii:

*student@9f22d9128c03:~$ sudo mn*

*\*\*\* Error setting resource limits. Mininet's performance may be affected.*

*\*\*\* Creating network*

*\*\*\* Adding controller*

*\*\*\* Adding hosts:*

*h1 h2*

*\*\*\* Adding switches:*

*s1*

*\*\*\* Adding links:*

*(h1, s1) (h2, s1)*

*\*\*\* Configuring hosts*

*h1 h2*

*\*\*\* Starting controller*

*c0*

*\*\*\* Starting 1 switches*

*s1 ...*

*\*\*\* Starting CLI:*

*mininet>*

**4.2 OpenVSwitch**

Program OpenVSwitch jest najpopularniejszą open sourcową implementacją przełącznika SDN. Za pomocą interfejsów wirtualnych pozwala on na tworzenie przełączników SDN. Mininet wykorzystuje OpenVSwitch podczas tworzenia sieci. W celu sprawdzenia poprawności działania przełącznika wydaj polecenie:

*sudo ovs-vsctl show*

W wyniku powinna pojawić się wykorzystywana wersja:

*student@9f22d9128c03:~$ sudo ovs-vsctl show*

*a4332235-127f-424b-9563-1319e7ba45f7*

*ovs\_version: "2.7.1"*

**4.3 Floodlight**

Jako implementację sterownika SDN wybrano projekt Floodlight. Jest to sterownik napisany z wykorzystaniem języka Java. Jego źródła zostały pobrane do katalogu */home/student/floodlight*. Ze źródeł zbudowano projekt dla środowiska *eclipse* (dzięki temu możliwe jest zaimportowanie). Aby zaimportować projekt floodlight do eclipse wykonaj następujące kroki:

1. Uruchom eclipse poleceniem *eclipse*
2. Pozostaw domyślną lokalizacje przestrzeni roboczej
3. Wybierz File → Import → General → Existing Projects into Workspace
4. Wybierz katalog */home/student/floodlight*
5. Zaimportuj projektem

Aby była możliwość uruchomienia kontrolera, należy dodać konfigurację startową. Aby to zrobić wykonaj następujące kroki:

1. Z menu górnego wybirze Run → Run Configurations…
2. Kliknij dwukrotnie na Java Application
3. Zmień nazwę na Floodlight
4. Jako Main class wybierz: net.floodlightcontroller.core.Main
5. Zatwierdź Apply i wyjdź Close

Aby uruchomić kontroler:

1. Zaznacz projekt floodlight
2. Naciśnij
3. Wybierz stworzoną wcześniej konfigurację (Floodlight)
4. Zatwierdź ok

W konsoli powinny pojawić się logi kontrolera:

2017-07-25 09:55:00.834 INFO [n.f.c.m.FloodlightModuleLoader] Loading modules from src/main/resources/floodlightdefault.properties

2017-07-25 09:55:01.46 WARN [n.f.r.RestApiServer] HTTPS disabled; HTTPS will not be used to connect to the REST API.

2017-07-25 09:55:01.46 WARN [n.f.r.RestApiServer] HTTP enabled; Allowing unsecure access to REST API on port 8080.

2017-07-25 09:55:01.46 WARN [n.f.r.RestApiServer] CORS access control allow ALL origins: true

2017-07-25 09:55:01.303 WARN [n.f.c.i.OFSwitchManager] SSL disabled. Using unsecure connections between Floodlight and switches.

2017-07-25 09:55:01.303 INFO [n.f.c.i.OFSwitchManager] Clear switch flow tables on initial handshake as master: TRUE

2017-07-25 09:55:01.303 INFO [n.f.c.i.OFSwitchManager] Clear switch flow tables on each transition to master: TRUE

2017-07-25 09:55:01.311 INFO [n.f.c.i.OFSwitchManager] Setting 0x1 as the default max tables to receive table-miss flow

2017-07-25 09:55:01.400 INFO [n.f.c.i.OFSwitchManager] OpenFlow version OF\_15 will be advertised to switches. Supported fallback versions [OF\_10, OF\_11, OF\_12, OF\_13, OF\_14, OF\_15]

2017-07-25 09:55:01.401 INFO [n.f.c.i.OFSwitchManager] Listening for OpenFlow switches on [0.0.0.0]:6653

2017-07-25 09:55:01.401 INFO [n.f.c.i.OFSwitchManager] OpenFlow socket config: 1 boss thread(s), 16 worker thread(s), 60000 ms TCP connection timeout, max 1000 connection backlog, 4194304 byte TCP send buffer size

2017-07-25 09:55:01.402 INFO [n.f.c.i.Controller] ControllerId set to 1

2017-07-25 09:55:01.402 INFO [n.f.c.i.Controller] Shutdown when controller transitions to STANDBY HA role: true

2017-07-25 09:55:01.402 WARN [n.f.c.i.Controller] Controller will automatically deserialize all Ethernet packet-in messages. Set 'deserializeEthPacketIns' to 'FALSE' if this feature is not required or when benchmarking core performance

2017-07-25 09:55:01.403 INFO [n.f.c.i.Controller] Controller role set to ACTIVE

2017-07-25 09:55:01.468 INFO [n.f.l.i.LinkDiscoveryManager] Link latency history set to 10 LLDP data points

2017-07-25 09:55:01.470 INFO [n.f.l.i.LinkDiscoveryManager] Latency update threshold set to +/-0.5 (50.0%) of rolling historical average

2017-07-25 09:55:01.472 INFO [n.f.t.TopologyManager] Path metrics set to LATENCY

2017-07-25 09:55:01.472 INFO [n.f.t.TopologyManager] Will compute a max of 3 paths upon topology updates

2017-07-25 09:55:01.482 INFO [n.f.f.Forwarding] Default hard timeout not configured. Using 0.

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Default idle timeout set to 5.

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Default table ID not configured. Using 0x0.

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Default priority not configured. Using 1.

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Default flags will be set to SEND\_FLOW\_REM false.

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Default flow matches set to: IN\_PORT=true, VLAN=true, MAC=true, IP=true, FLAG=true, TPPT=true

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Default detailed flow matches set to: SRC\_MAC=true, DST\_MAC=true, SRC\_IP=true, DST\_IP=true, SRC\_TPPT=true, DST\_TPPT=true

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Not flooding ARP packets. ARP flows will be inserted for known destinations

2017-07-25 09:55:01.483 INFO [n.f.f.Forwarding] Flows will be removed on link/port down events

2017-07-25 09:55:01.484 INFO [n.f.s.StatisticsCollector] Statistics collection disabled

2017-07-25 09:55:01.484 INFO [n.f.s.StatisticsCollector] Port statistics collection interval set to 10s

2017-07-25 09:55:01.545 INFO [o.s.s.i.SyncManager] [1] Updating sync configuration ClusterConfig [allNodes={1=Node [hostname=192.168.1.100, port=6642, nodeId=1, domainId=1], 2=Node [hostname=192.168.1.100, port=6643, nodeId=2, domainId=1]}, authScheme=CHALLENGE\_RESPONSE, keyStorePath=/etc/floodlight/key2.jceks, keyStorePassword is set]

2017-07-25 09:55:01.672 INFO [o.s.s.i.r.RPCService] Listening for internal floodlight RPC on 0.0.0.0/0.0.0.0:6642

2017-07-25 09:55:01.885 INFO [o.r.C.I.Server] Starting the Simple [HTTP/1.1] server on port 8080

2017-07-25 09:55:01.886 INFO [org.restlet] Starting net.floodlightcontroller.restserver.RestApiServer$RestApplication application

2017-07-25 09:55:04.702 INFO [n.f.j.JythonServer] Starting DebugServer on :6655

Sprawdźczy nie ma w nich informacji o błędach!!!

**5. Podsumowanie**

Obraz Dockera zawiera niezbędne narzędzia do wykonania zadań laboratoryjnych. Po przejściu ninejszej instrukcji student powinien potrafić posługiwać się kontenerem Dockera, uruchomić środowisko emulacyjne mininet oraz sterownik Floodlight.