Batuhan AVLAYAN – batuhanavlayan@etu.edu.tr

Github Repo Link: https://github.com/bavlayan/Usom-IP-Blocker-using-P4

Ödev 2 – P4 destekli ağ anahtarlarının IP paketlerinin işlerken kullanıcı tarafından belirlenecek IP adreslerini engelleyecek şekilde programlanması

Projenin amacı, resimdeki topolojide bulunan switchleri P4 desteğini kullanarak, usom tarafından zararlı olduğu belirlenmiş yaklaşık 130000 üzerinde web sitesine giden tüm paketleri engellemek ve switchleri konfigure etmek için tablo kurallarını oluşturan uygulamayı geliştirmek.

Not: Bu uygulamada sadece S1 switch'i için konfigurasyon ayarlaması yapılacaktır.

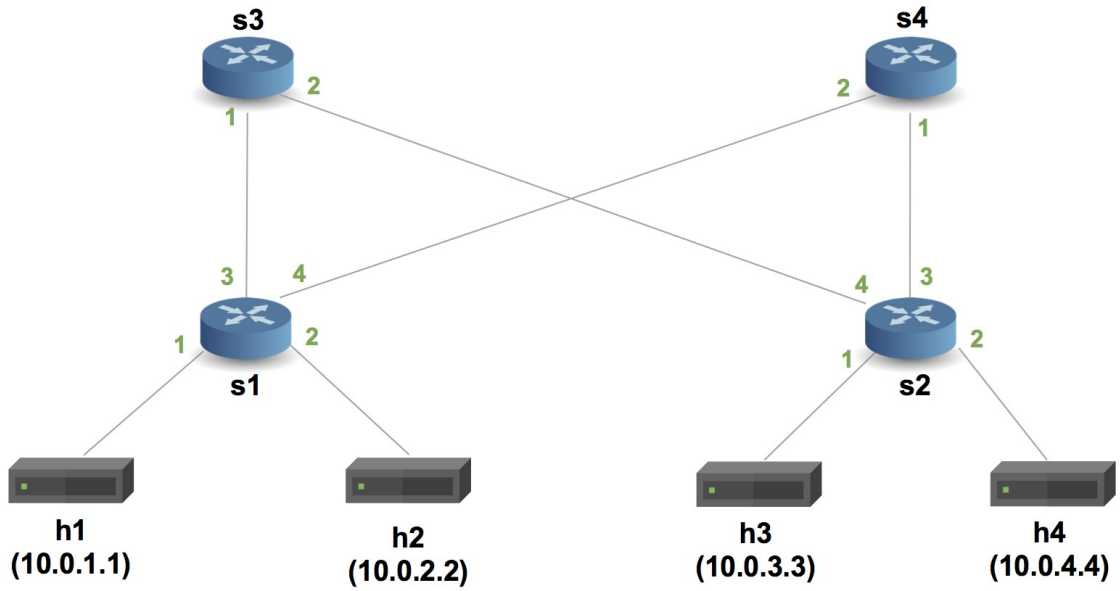

Figure 1: Topoloji

Topolojide toplam 4 switch ve 4 host bulunmaktadır. Zararlı url'lere giden paket engelleme işlemi S1 olan switch üzeride gerçekleşmektedir. Yalnız h1 ve h2 hostlarından bu adreslere gidecek olan ipv4 paketlerinin engellmesi yapılacaktır.

- Proje aşağıdaki dosyalardan oluşmaktadır:
  1 – blocked_url.json
  2 – BlockedUrl.py
  3 – Constants.py
  4 – P4SwitchJsonCreator.py
  5 – s1-runtime.json
  6 – UsomUrlHelper.py

1. blocked_url.json

Yasaklı ***aktif*** url bilgilerinin json formatında tutulduğu dosya. Program çalışırken yasaklı url bilgilerini buradan alarak kullanmaktadır. Json formatı aşağıdaki gibidir. Json objesinde url ismi, url'in ip adresi ve url'in aktif olup olmadığını tutan nitelikler bulunmaktadır.

{ "url_name" : "e-devlet-online.net", "ip": "99.83.154.118", "is_active":"true" }

2. BlockedUrl.py

BlockedUrl sınıfı yasaklanmış url bilgilerinin tutulduğu özellikleri içermektedir. **__eq__** ve **__hash__** metodları override edilerek ***aynı ip adresine sahip olan farklı url*** adreslerini set'e eklerken tekilleştirilmek için kullanılmaktadır.

```
class BlockedUrl():
    def __init__(self, url_name, ip, is_active):
        self.url_name = url_name
        self.ip = ip
        self.is_active = is_active

    def __eq__(self, other):
        return self.ip == other.ip

    def __hash__(self):
        return hash(self.ip)
```

Figure 2: BlockedUrl Class

3. Constants.py

Tüm program içinden kullanılan sabit değerlerin tutulduğu python dosyası.

4 – P4SwitchJsonCreator.py

Figure-1'de çizilen toplojide bulunan her bir switch için p4 derleyicisi tarafından kullanılan json formatında tablo kurallarının ve switch ayarlarının içerdeği bir json dosyası bulunmaktadır. P4 derleme sırasında bu json dosyasını okuyarak switchleri yapılandırıyor ve aynı zamanda ilgili oluşturulmuş tablo kurallarını switch'e tanımlıyor. P4SwitchJsonCreator.py dosyasında input olarak verilen switch ayarlının içerdiğini json dosyasına UsomUrlHelper.py tarafından oluşturulan blocked_url.json dosyasını okuyarak bu switch ayar dosayını tekrardan oluşturuyor. B***urada ki en önemli nokta switch kuralları tekrardan oluşturulmuyor. Sadece blocke_url.json dosyasında bulunan ip adresleri kural olarak ekleniyor.***

P4SwtichJsonCreator.py dosyası 3 fonksiyondan oluşmaktadır.

- **create_dropped_table_entries:** Bu fonksiyonda, usom_blocked_url_list dizisinde bulunan her bir bloklanmış url'lerin ip adresini alıp, json formatında switch için drop kuralını oluşturmaktadır. (Figure 3)

```
def create_dropped_table_entries():
    try:
        if switch_settings_dictionary is not None and usom_blocked_url_list is not None:
            current_table_entires = switch_settings_dictionary['table_entries']
            for usom_blocked_url in usom_blocked_url_list:
                blocked_ip = usom_blocked_url['ip']
                drop_object = {
                    'table': 'MyIngress.ipv4_lpm',
                    'match': {
                        'hdr.ipv4.dstAddr': [blocked_ip, 32]
                    },
                    'action_name' : 'MyIngress.drop',
                    'action_params' : {}
                }
                current_table_entires.append(drop_object)
            switch_settings_dictionary['table_entries'] = current_table_entires
            return switch_settings_dictionary
    except:
        print("An error occured in create_dropped_table_entries method")
        sys.exit()
```

Figure 3: create_dropped_table_entries function

- **load_json_by_file_name:** Bu fonksiyon, parametre olarak verilen dosya ismini alarak, json dosyasını okur ve parametre olarak sözlük objesi döner.
- **write_json_file:** Bu fonksiyon, parametre olarak verilen sözlük objesi ve dosya ismini alarak, o isimde json dosyası oluşturur.

5. s1-runtime.json

Switch ayarlarının json formatında bulunduğu dosyadır. Bu switch için oluşturulacak kurallar table_entries bölümü (array) altına eklenmektedir. (Figure 4: Örnek switch kurallarının ve ayarlarının olduğu json dosyası)

Figure 4: Örnek Switch kurallarının ve ayarlarının olduğu json formatı

6 – UsomUrlHelper.py

https://www.usom.gov.tr/url-list.txt adresinden çekilen adreslerin ip'lerinin belirlenmesi işlemlerini yapılmaktadır. *get_blocked_urls_from_usom* fonksiyonu usom adresinden alınan zaralı bağlantı adreslerinin parse işlemleri yapılmaktadır.

```
def __get_blocked_urls_from_usom(self):
    try:
        response = urllib.request.urlopen(self.usom_url, timeout=3)
        for url in response:
            url_name = url.decode("utf-8").rstrip("\n")
            is_ip = self.__check_ip(url_name)
            if is_ip is True:
                continue
            blocked_url = BlockedUrl(url_name, '', False)
            self.blocked_url_list.append(blocked_url)
    except:
        print("Url not open")
        sys.exit()
```

Figure 5: get_blocked_urls_from_usom function

get_ip_from_url fonksiyonu ile url adreslerinin ip adresi tespit edilmektedir.

```
def __get_ip_from_url(self, blocked_url_list):
    for blocked_url in blocked_url_list:
        try:
            ip_address = socket.gethostbyname(blocked_url.url_name)
            blocked_url.is_active = True

            if not ip_address:
                print(Fore.YELLOW + blocked_url.url_name)
                blocked_url.is_active = False
                self.blocked_url_list.remove(blocked_url)

            if self.__is_ip_private(ip_address):
                blocked_url.is_active = False
                self.blocked_url_list.remove(blocked_url)

            blocked_url.ip = ip_address
            print(Fore.GREEN + "Url:" + blocked_url.url_name + " Ip:" + blocked_url.ip)
        except:
            blocked_url.is_active = False
            print(Fore.RED + "Url:" +blocked_url.url_name + " IP Address not found")
            self.blocked_url_list.remove(blocked_url)
            continue
```

Figure 6: get_ip_from_url function

set_ip fonksiyonu ile oluşturulmuş olan oluşturalan blocked_url_list array'inde bulunan url'lerin ip set edilme işlemi yapılmaktadır. blocked_url_list array length'i çok büyük olduğundan burada ip et edilme işlemi thread ile yapılmakatadır.

```
def __set_ip(self):
    try:
        thread_arr = []
        loop_last_index = int(len(self.blocked_url_list) / self.thread_count)
        first_index = 0
        for i in range(self.thread_count):
            partial_blocked_list = self.blocked_url_list[first_index:loop_last_index]
            first_index = loop_last_index
            loop_last_index = loop_last_index * 2
            t = threading.Thread(target=self.__get_ip_from_url, args=(partial_blocked_list,))
            thread_arr.append(t)

        for t in thread_arr:
            t.start()

        for t in thread_arr:
            t.join()
    except:
        print("Getting error in set ip function")
```

Figure 7: set_ip function

check_ip fonksiyonu ile Usom'dan alınan url listesinde bulunan ip adreslerinin tespit edilmesi işlemi yapılmakatadır.

```
def __check_ip(self, url):
    ip_regex = "^((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])\.){3}(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])$"
    if(re.search(ip_regex, url)):
        return True
    return False
```

Figure 8: check_ip function

is_ip_private fonksiyonu bir ip adresinin özel ip blokları arasında olup olmadığını tespit etmektedir.

```
def __is_ip_private(self, ip):
    priv_lo = re.compile("^127\.\d{1,3}\.\d{1,3}\.\d{1,3}$")
    priv_24 = re.compile("^10\.\d{1,3}\.\d{1,3}\.\d{1,3}$")
    priv_20 = re.compile("^192\.168\.\d{1,3}.\d{1,3}$")
    priv_16 = re.compile("^172.(1[6-9]|2[0-9]|3[0-1]).[0-9]{1,3}.[0-9]{1,3}$")
    res = priv_lo.match(ip) or priv_24.match(ip) or priv_20.match(ip) or priv_16.match(ip)
    return res is not None
```

Figure 9: is_ip_private

create_json_file fonksiyonu *blocked_url_list* array'in json dump edilerek *blocked_url.json* dosyasını oluşturmaktadır. Burada kritik olan nokta ip adreslerinin tekilleştirilmesi. *blocked_rul_list* dizisi **set** objesine dönüştürülerek ip'ye **(Figure 2'de BlockedUrl sınfında bulunan __hash__ ve __eq__ metodu sayesinde)** göre tekilleştirme işlemi yapılmaktadır.

```
def create_json_file(self):
    self.__get_blocked_urls_from_usom()
    self.__set_ip()
    result = list(set(self.blocked_url_list))
    try:
        with open(Constants.BLOCKED_URL_JSON_FILE_NAME, 'w') as json_file:
            json.dump(result, json_file, default=vars, indent=2)
            print(Fore.GREEN + "blocked_url.json file was created...")
    except:
        print(Fore.RED + "An error occured when creating " + Constants.BLOCKED_URL_JSON_FILE_NAME + " json file")
```

Figure 10: create_json_file function

PROGRAMIN ÇALIŞTIRILMASI VE TEST SONUÇLARI

1. Programın Çalıştırılması

Topolojinin oluşturulması ve mininet üzerinde simule edilmesi için p4'ün hazır kurulu olduğu sanal makina ova image olarak indirilip VirtualBox üzerinde çalıştırılmalıdır.

VMBox .ova image download link:
https://drive.google.com/file/d/1ZkE5ynJrASMC54h0aqDwaCOA0I4i48AC/view

Mavi adımlar sanal makina üzerinden yapılmalıdır.

Simule işleminin başarılı bir şekilde yapılması için aşağıdaki adımlar sırası ile uygulanmalıdır. ***Farklı sırada yapılması durumunda simule işlemi başarısız olacaktır.***

1. ***python3 UsomUrlHelper.py*** – komutu çalıştırılarak ***blocked_url.json*** dosyası oluşturulmalı.
2. Sanal makina içerisinde bulunan ***/home/p4/tutorials/exercies/basic/pod-topo*** dizni altında bulunan ***s1-runtime.json*** dosyası program dosyalarımızın olduğu dizine kopyalanmalıdır.
3. ***python3 P4SwitchJsonCreator.py*** – komutu çalıştırılarak ***s1-runtime.json*** dosyasına ilgili drop kurallarını oluşturuyoruz.
4. ***s1-runtime.json*** dosyasını kontrol ettiğimizde aşağıdaki resimde bulunan drop kurallarını eklendiğini göreceğiz.

```
{
  "table": "MyIngress.ipv4_lpm",
  "match": {
    "hdr.ipv4.dstAddr": ["172.10.12.12", 32]
  },
  "action_name": "MyIngress.drop",
  "action_params": {
  }
}
```

Figure 11: Drop Rule

5. ***s1-runtime.json*** dosyasını tekrar ***/home/p4/tutorials/exercies/basic/pod-topo*** altına kopyalıyoruz.
6. **cd /home/p4/tutorials/exercises/basic –** komutu dizinine geçiyoruz
7. **make clean** – komutu ile p4 projesi tüm build sonu oluşmuş geçmiş dosyaları temizliyoruz.
8. **make run** – komutu ile topolojimizi mininet ortamında simule edebilmek için çalıştırıyoruz. Eğer ***mininet> prompt*** konsolu geldiğinde topolojimiz başarılı bir şekilde simule edilmiş ve s1 switch için kurallarımız eklenmiş olacaktır. **pingall** komutu ile test yapabiliriz

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet>
```

Figure 12: mininet prompt

2. S1 Swtich drop kurallarının table_dump komutu ile incelenmesi

1. **simple_switch_CLI --thrift-port 9090** komutu ile S1 switch konsoluna bağlanıyoruz.

```
p4@p4:~$ simple_switch_CLI --thrift-port 9090
Obtaining JSON from switch...
Done
Control utility for runtime P4 table manipulation
RuntimeCmd:
```

Figure 13: S1 Switch CLI

2. **table_dump MyIngress.ipv4_lpm** komutu ile S1 Switch üzerinden girilmiş olan kuralları görebiliriz. Örnek olarak 2 drop kuralı aşağıdaki resimde görülebilir.

```
Dumping entry 0x62
Match key:
* ipv4.dstAddr       : LPM       c31432fc/32
Action entry: MyIngress.drop -
**********
Dumping entry 0x63
Match key:
* ipv4.dstAddr       : LPM       ac43b79b/32
Action entry: MyIngress.drop -
**********
```

Figure 14: Drop Rules in S1 Switch

IP adresleri hexadecimal formatta çevrilmektedir. ac43b79b decimal olarak 172.67.183.155 ip'sine karşılık gelmektedir.

```
{
  "url_name": "b1imcelletleeyuklee.com",
  "ip": "172.67.183.155",
  "is_active": true
},
```

Figure 15: Blocked Url Sample