PRÉSENTATION PROJET SDN

DÉTECTION ET MITIGATION D'ATTAQUE DDOS

PAR CONTRÔLEUR OPENFLOW

ENVIRONNEMENT: MININET ,QEMU-KVM, PYTHON ,POX

MODULE :PRIVACITÉ ET SÉCURITÉ DES DONNÉES

ENSEIGNANT : MR HOUARNOUGHI HAMZA

PAR : MAHAMAN BACHIR ATTOUMAN OUSMANE

M2 CDSI / UNIVERSITÉ POLYTECHNIQUE HAUTS-DE-FRANCE

PLAN

1. Présentation de l’environnement
2. Automatisation de la gestion des Vms
3. Machines hôtes mininet
4. Controlleur pox
5. Perspectives

PRÉSENTATION DE L'ENVIRONNEMENT

AUTOMATISATION

- python
  - Mettre en marche ou arrêter Toutes les Vms
  - Ou mettre en marche/arrêter une Vm spécifique
  - Voir les machines actives/inactives

Éviter les longues commandes répétitives en créant des Alias

Connexion SSH aux Vms (Par clé asymétrique)

Quand faire la requête de connexion ?

1. Récupérer le nom du Bridge du domaine avec libvirt

Faire :

Lire la table ARP

Récupérer les adresses IP des lignes contenant le nom du pont

Tant que Nombre d'IP != Nombre de domaine mis en marche

MININET HOSTS

```
server.cmd("nohup python -m SimpleHTTPServer 8888 >minilog/server.log &")
print("[+] Web server is listenning")


g1.cmd("nohup python script/gen1 > minilog/g1.log &")
print("[+] traffic generator 1 is on")

g2.cmd("nohup python script/gen2 > minilog/g2.log &")
print("[+] traffic generator 2 on")

client.cmd("nohup python3 script/client.py 10.0.0.11:8888 > minilog/client.log &")
print("[+] client up")
time.sleep(2)

dos.cmd("nohup python3 script/httpflood.py > minilog/attacker.log &")
print("[*] DOS attack launched")



CLI(net)
```

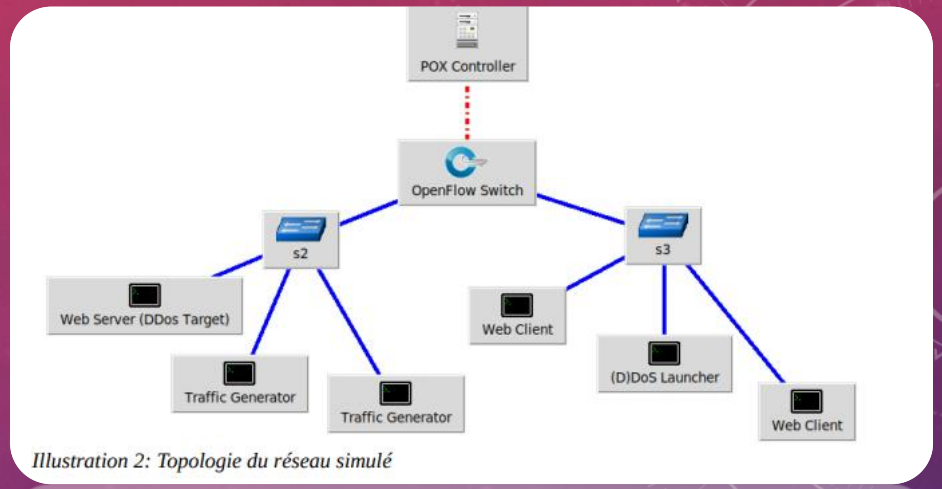

Illustration 2: Topologie du réseau simulé

Création de l'architecture :

- deux switch legacy
- un switch openflow
- adresses statiques
- Contrôleur distant

2 secondes d'attente avant de lancer l'attaque

Client : effectue une requête HTTP chaque seconde et calcule le temps de réponse

Attaquant : Réalise des requêtes HTTP vers le serveur , avec des Threads

Traffic generator : Traffic TCP aléatoire avec scapy , 1 paquet / 0.8 secondes

SWITCH POX

```
    packet = event.parsed
    #Ajout d'une entrée à la table de commutation
    table[packet.src] = event.port
    # essai de lecture du port de destination corespondant
    #à la mac de destination
    dst_port = table.get(packet.dst)
    if str(packet.src) in blacklist:
        sendpacket(event,port = 0)
    #blacklist contenant les ports physiques à ignorer

    elif packet.dst.is_multicast:
        sendpacket(event,of.OFPP_FLOOD)
    # gestion des Broadcast , à limiter
    elif str(packet.dst) not in table:
        sendpacket(event,of.OFPP_FLOOD)
    else:
    #creation et transmission d'une ligne de table de switch
        dst_port = packet.dst
        msg = of.ofp_flow_mod()
        msg.idle_timeout = 10
        msg.hard_timeout = 30
        msg.actions.append(of.ofp_action_output(port = dst_port))
        event.connection.send(msg)
```

DOS ATTACK

Création d'un dictionnaire , contenant le nombre de paquets par flux en T secondes :

Exemple :

Source = port 3 ; destination = AA:BB:CC:DD

{'3;AA:BB:CC:DD' : 400}

Détection par seuil de nombre de paquet , issus du port physique vers une destination unique

En cas de détection , nouveau flow qui drop tout les paquets issus de ce port pendant X temps

PERSPECTIVES :

- CALCULER LE SEUIL PAR UN ALGORITHME
- APPRENDRE L’ ARCHITECTURE (LE CONTRÔLEUR) EN TROUVANT LES TYPES DE SERVEURS ET LE PROFIL DES HÔTES