Université de Bordeaux — année 2019-2020

Master CSI/Master Info. — Sécurité des réseaux

TD - Mise en place d'un système de prévention d'intrusions

Le but de ce TP est de mettre en place un système de détection d'intrusions. Ce mécanisme sera mis en place en utilisant le logiciel *suricata*.

La topologie réseau correspondante peut être obtenue en lançant le script de démarrage `/net/stockage/aguermou/SR/TP/6/qemunet.sh` en lui fournissant la description de la topologie réseau à l'aide de l'option `-t` ainsi que l'archive contenant la configuration initialte des machines à l'aide de l'option `-a`. Ceci revient à lancer les commandes suivantes :

```
cd /net/stockage/aguermou/SR/TP/6/; ./qemunet.sh -x -t topology -a archive_tp6.tgz
```

Le système de prévention d'intrusions doit être mis en place sur immortal.

1. Commencer la configuration de *suricata*. Il faut tout d'abord définir les options de lancement de `suricata`. Nous allons donc spécifier ces dernières en surchargeant le script de lancement `systemd` de `suricata`.

   ```
   # mkdir /etc/systemd/system/suricata.service.d

   # cat > /etc/systemd/system/suricata.service.d/suricata.conf

   [Service]
   ExecStart=
   ExecStart=/usr/bin/suricata -D -q 0 -c /etc/suricata/suricata.yaml\
              --pidfile /var/run/suricata.pid


   # systemctl daemon-reload
   ```

   Il faut surtout remarquer l'option `-q` qui permet de dire à suricata de gérer lui même les paquets mis de coté par le firewall via la cible `NFQUEUE`.

2. Éditer le script `/etc/init.d/ma-config.sh` pour y spécifier le trafic à envoyer vers `suricata`. Dans le cas suivant, nous allons envoyé l'intégralité du trafic transitant par le firewall vers `suricata` (ce scénario n'étant pas réaliste en production).

   ```
   iptables -A FORWARD -j NFQUEUE
   ```

3. Nous allons ajouter un fichier décrivant les régles que nous allons créer pour `suricata`. Pour ce faire, il faut tout d'abord créer un ficher (`myrules.rules` par exemple).

   ```
   # touch /etc/suricata/rules/myrules.rules
   ```

   Puis dire à `suricata` de le charger en spécifiant ce ceci dans le fichier de configuration `/etc/suricata/suricata.yaml`.

   ```
   rule-files:
    - myrules.rules
   ```

4. Nous allons définir une première règle (dans le fichier que vous venez de créer) qui consiste à emettre une alerte lorsqu'un paquet `icmp` transite par immortal.

```
alert icmp any any -> any any (msg:"SURICATA my ICMP alert"; sid:22400321;)
```

Il faut alors redémarrer le service via `service suricata restart`, puis générer un traffic `icmp` transitant par immortal et enfin consulter le fichier contenant les alertes : `/var/log/suricata/fast.log`. Il est à noter que l'on peut interdir dynamiquement le traffic correspondant à l'alerte en remplaçant le mot clé `alert` par le mot clé `drop`.

5. Nous allons maintenant nous intéresser à un scénario un peu plus complexe dans lequel une alerte est générée si trois tentatives de connexion ont échoué au sein d'une session telnet. Pour ce faire, nous allons utiliser la fonctionnalité `flowint` qui permet de manipuler les compteurs nécessaires. Ce qui est demandé peut être fait avec trois règles qui permettent dans l'ordre, d'intialiser, d'incrémenter et de tester la valeur du compteur.

```
#initialiser le compteur
alert tcp-pkt any any <> any 23 (msg:"Init Failing logins counter";\
    content:"Login incorrect";flow:established,from_server;\
    flowint: username, notset;\
    flowint:username, =, 1; noalert; sid:1;)

#incrémenter le compteur
alert tcp-pkt any any <> any 23 (msg:"Inc Failed Logins";\
    content:"Login incorrect";flow:established,from_server;\
    flowint: username, isset;\
    flowint:username, +, 1; noalert; sid:2;)

#générer une alerte lorsque le compteur est supérieur à 2
alert tcp-pkt any any <> any 23 (msg:"More than 3 Failed Logins!";\
    content:"Login incorrect"; flow:established,from_server; \
    flowint: username, isset;\
    flowint:username, >, 2; sid:3;)
```

Générer le trafic correspondant et vérifier que l'alerte est bien émise (en consultant le fichier `/var/log/suricata/fast.log`.

6. Nous allons utiliser maintenant une foncitionnalité avancée de `suricata` qui permet de définir le comportement d'une règle par le biais de l'exécution d'un script `lua`. Ce script sera chargé de dire si l'alerte doit être émise ou non. Un exemple pratique de l'utilisation de ce mécanisme consiste à analyser un protocole de haut-niveau pour détecter un comportement pré-défini. Nous allons nous intéresser à une connexion `TLS` et nous souhaitons émettre une alerte lorsque le certificat utilisé est auto signé par la machine qui le posséde. Il faut dans un premier être en mesure de récupérer les certificats de manière automatique au niveau de `suricata`. Pour ce faire, il faut dans un premier temps décommenter/modifier les lignes suivantes dans `/etc/suricata/suricata.yaml` :

```
  # a line based log of TLS handshake parameters (no alerts)
- tls-store:
  enabled: yes  # Active TLS certificate store.
  certs-log-dir: certs # directory to store the certificates
```

Il faut ensuite écrire la règle qui permet qui fait référence au script qui va analyser le certificat et l'ajouter à notre fichier de règles.

```
alert tls any any -> any any (msg:"SURICATA TLS Self Signed Certificate"; \
  flow:established; luajit:<nom_script_lua>; \
  tls.store; classtype:protocol-command-decode; sid:99666321; rev:1;)
```

Enfin, il faut copier le script `tls.lua` se trouvant dans le dossier `/net/stockage/aguermou/SR/TP/6/files/tls.lua` pour le mettre dans le dossier `/etc/suricata/rules`. Pour effectuer le test, des certificats ont été préparés pour nile et sont disponibles dans l'archive

`nile-certs.tgz` se trouvant dans le même dossier que le script `tls.lua`. L'idée est alors de mettre `nile` en écoute sur un port donné en utilisant l'outil `gnutls-serv` et de s'y connecter depuis opeth par exemple en utilisant l'outil `gnutls-cli`. Vérfier que l'alerte est bien générée.

Remarque : `Suricata` ne supportant pas totalement `TLS1.3`, il est nécessaire de forcer l'utilisation d'une version antérieure (`TLS1.2` par exemple) :

```
gnutls-serv --priority SECURE256:-VERS-TLS-ALL:+VERS-TLS1.2 ...
```