Ochrona danych i systemów
Milestone 2

Temat 1: Aktywny Firewall

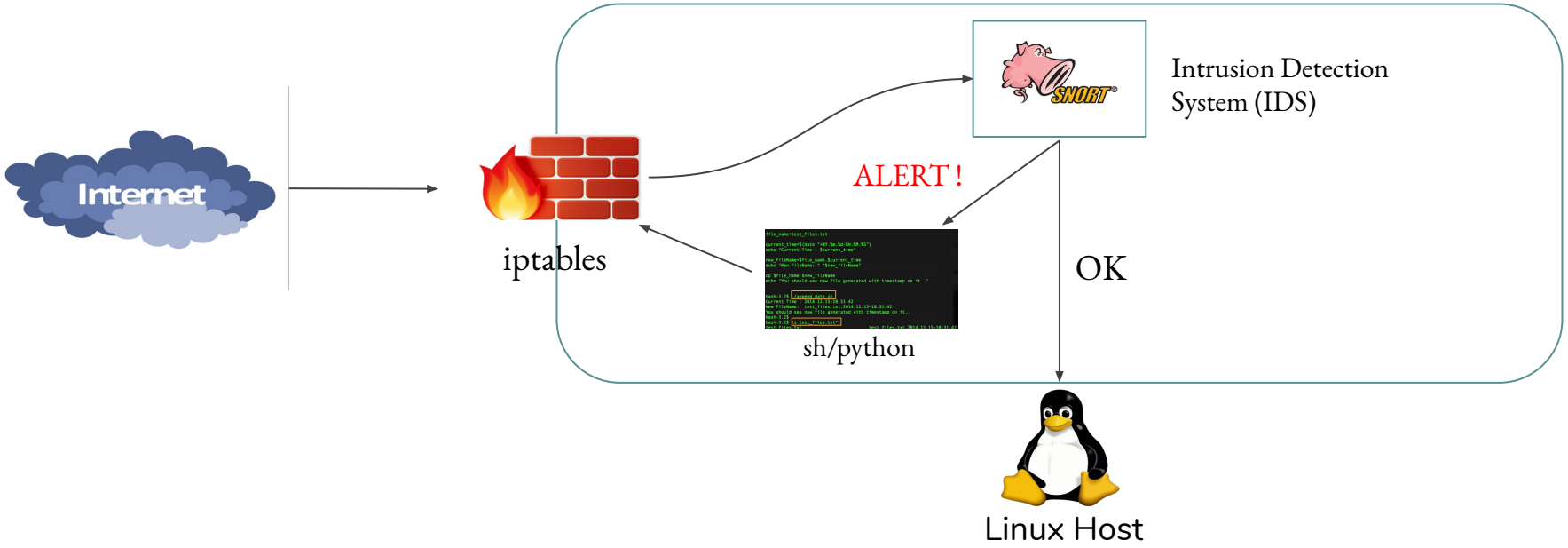
Skład zespołu:

- Alicja Uzar
- Daniel Mynarski
- Dominik Wróbel

Plan prezentacji

1. Przypomnienie architektury projektu
2. Omówienie realizacji elementów architektury projektu
    a. Wykrycie ataku - Snort
        i. Konfiguracja
        ii. Reguły
    b. Obrona przed atakami - skrypty i iptables
        i. Omówienie przykładowego ataku
        ii. Obrona przed atakiem

Architektura



Snort - zależności i konfiguracja

- **Instalacja:**

```
sudo apt-get update
# instalacja zaleznosci dla snort:
sudo apt-get install libpcap-dev bison flex
# instalacja snort:
sudo apt-get install snort
# sprawdzenie instalacji:
man snort
```

Biblioteka przechwytywanie pakietów dla Linux

- **Pliki konfiguracyjne:**

Reguły filtracji pakietów

Konfiguracja snort

```
dominik@dominik-VirtualBox:/etc/snort$ ls
classification.config  gen-msg.map       rules        snort.debian.conf  unicode.map
community-sid-msg.map  reference.config_ snort.conf  threshold.conf
```

Snort - tryby i przepływ pakietów

Packet Sniffer | Packet Logger | **Intrusion Detection System**

SNORT

Pakiety

1. Określa protokół

2. Dopasowuje dane do dopuszczalnego zachowania dla pakietów ich protokołu

3. Może generować alerty na podstawie budowy pakietów (np. zbyt długi pakiet)

Dekoder

1. Analizują różne cechy charakterystyczne pakietów, wykrywają charakterystyczne ataki

2. Mogą również przygotowywać strumień danych dla kolejnych modułów IDS

3. Np. *portscan* - wykrywa próby skanowania portów

Preprocesory (wiele, opcjonalne)

1. Badanie pakietów metodą analizy sygnatur:
   - Adresy IP
   - Numery portów
   - Rodzaje i typy pakietów
   - itd.

2. Korzysta z plików z rozszerzeniem *.rules* do dopasowania pakietów

Analiza sygnatur (detection engine)

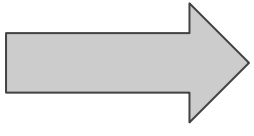
1. Różne możliwości reakcji:
- zgłaszanie alertów
- zapis do pliku
- przerwanie połączenia TCP
- inne

Moduł reakcji na ataki

Plik snort.conf

- Wyłączenie wszystkich zdefiniowanych domyślnie reguł aby uniknąć analizowania nieistotnych z punktu widzenia projektu zdarzeń

```
#include $RULE_PATH/app-detect.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules
#include $RULE_PATH/browser-chrome.rules
#include $RULE_PATH/browser-firefox.rules
#include $RULE_PATH/browser-ie.rules
#include $RULE_PATH/browser-other.rules
#include $RULE_PATH/browser-plugins.rules
#include $RULE_PATH/browser-webkit.rules
include $RULE_PATH/chat.rules
#include $RULE_PATH/content-replace.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/experimental.rules
#include $RULE_PATH/exploit-kit.rules
include $RULE_PATH/exploit.rules
#include $RULE_PATH/file-executable.rules
#include $RULE_PATH/file-flash.rules
#include $RULE_PATH/file-identify.rules
#include $RULE_PATH/file-image.rules
#include $RULE_PATH/file-multimedia.rules
#include $RULE_PATH/file-office.rules
#include $RULE_PATH/file-other.rules
#include $RULE_PATH/file-pdf.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/icmp-info.rules
include $RULE_PATH/icmp.rules
```



```
#include $RULE_PATH/app-detect.rules
#include $RULE_PATH/attack-responses.rules
#include $RULE_PATH/backdoor.rules
#include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules
#include $RULE_PATH/browser-chrome.rules
#include $RULE_PATH/browser-firefox.rules
#include $RULE_PATH/browser-ie.rules
#include $RULE_PATH/browser-other.rules
#include $RULE_PATH/browser-plugins.rules
#include $RULE_PATH/browser-webkit.rules
#include $RULE_PATH/chat.rules
#include $RULE_PATH/content-replace.rules
#include $RULE_PATH/ddos.rules
#include $RULE_PATH/dns.rules
#include $RULE_PATH/dos.rules
#include $RULE_PATH/experimental.rules
#include $RULE_PATH/exploit-kit.rules
#include $RULE_PATH/exploit.rules
#include $RULE_PATH/file-executable.rules
#include $RULE_PATH/file-flash.rules
#include $RULE_PATH/file-identify.rules
#include $RULE_PATH/file-image.rules
#include $RULE_PATH/file-multimedia.rules
#include $RULE_PATH/file-office.rules
#include $RULE_PATH/file-other.rules
#include $RULE_PATH/file-pdf.rules
#include $RULE_PATH/finger.rules
#include $RULE_PATH/ftp.rules
#include $RULE_PATH/icmp-info.rules
```

Utworzenie pliku local.rules

- Budowa reguł

Action	Protocol	Source address	Source port	Direction	Destination address	Destination port	Rule options
- alert - log - pass - ...	- tcp - udp - icmp - ...	any	any	<> ->	$HOME_NET any	any 80	- msg - flags - detection_filter - count - seconds - sid - rev - ...

- Przykład detekcji LAND attack:

```
alert tcp any any -> any 80 (msg:"LAND ATTACK ALERT"; sameip; flags:S; sid: 1000003; rev:1;)
```

TCP Port Scanning

VirtualBox Kali Linux
10.0.2.4/24

nmap -sT 10.0.2.15

Snort + VirtualBox
Ubuntu
10.0.2.15/24

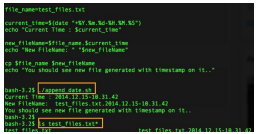
```
alert tcp any any -> any any (msg:"TCP PORT SCAN ALERT"; detection_filter:track by_src,
count 100, seconds 1; sid:1000004; rev:1;)
```

```
kali@kali:~$ nmap -sT 10.0.2.15
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-21 05:02 EDT
Nmap scan report for 10.0.2.15
Host is up (0.0014s latency).
All 1000 scanned ports on 10.0.2.15 are closed

Nmap done: 1 IP address (1 host up) scanned in 0.35 seconds
```

```
04/21-13:22:41.333464  [**] [1:1000004:1] TCP PORT SCAN ALERT [*
*] [Priority: 0] {TCP} 10.0.2.15:10009 -> 10.0.2.4:41568
04/21-13:22:41.333660  [**] [1:1000004:1] TCP PORT SCAN ALERT [*
*] [Priority: 0] {TCP} 10.0.2.4:47860 -> 10.0.2.15:1687
04/21-13:22:41.333669  [**] [1:1000004:1] TCP PORT SCAN ALERT [*
*] [Priority: 0] {TCP} 10.0.2.15:1687 -> 10.0.2.4:47860
```

Reakcja na TCP Port Scanning



Do iptables dodajemy regułę blokującą wysyłane pakiety od hosta o danym IP

iptables -A INPUT -s {IP} -j DROP

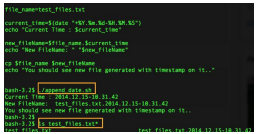
A - dodanie reguły do określonego łańcucha,
INPUT - wywoływany dla pakietów przybywających z sieci, przeznaczonych dla lokalnej maszyny **,**
-j DROP – usunięcie dopasowanych pakietów,
-**s {IP)** - adres docelowy o danym IP.

```
04/22-02:24:34.066386  [**] [1:1000004:1] TCP PORT SCAN ALERT [**] [Priority: 0] {TCP} 192.168.100.4:45326 -> 1
92.168.100.5:6839

Find attack: TCP PORT SCAN ALERT
Adding rule: -A INPUT -s 192.168.100.4/32 -j DROP
```

```
alicja@alicja-VirtualBox:~/Desktop$ sudo iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -s 192.168.100.4/32 -j DROP
```

Skrypty



firewall_start.sh

```
#!/bin/bash

sudo snort -d -l /var/log/snort/ -A console -c /etc/snort/snort.conf | sudo ./py/active-firewall.py
```

active_firewall.py - funkcja główna

```
def activeFirewall(line):
        for allert in attack_alerts:
                if allert in line:
                        print('Find attack: ' +  allert)
                        sourceIp = findIpAddress(line,0)
                        destIp = findIpAddress(line,1)
                        rule = getRule(sourceIp,'', '')
                        if not checkDangerIpBlocked(rule):
                                addRule(allert, sourceIp,rule)

for line in sys.stdin:
        activeFirewall(line)
```

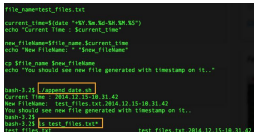
active_firewall.py - sprawdzenie czy istnieje już taka reguła w iptables

```
def checkDangerIpBlocked(rule):
        proc = subprocess.Popen(['iptables-save'], stdout=subprocess.PIPE)
        match = False
        for line in io.TextIOWrapper(proc.stdout, encoding="utf-8"):
                if not line:
                        break
                match = (rule in line)

                if match:
                        break

        return match
```

Skrypty



active_firewall.py - filtrowanie adresu IP

```
IP_REGEX = r"(\d{1,3}\.){3}\d{1,3}"
OPTIONAL_PORT_REGEX = r"(:\d{1,5})+"
IP_PORT_REGEX = IP_REGEX + OPTIONAL_PORT_REGEX

def findIpAddress(line, num):
        matched = re.search(IP_PORT_REGEX + " -> " + IP_PORT_REGEX, line);
        if matched is not None:
                ipport = matched.group().split(" -> ")[num]
                ip = ip = re.search(IP_REGEX, ipport).group()
                return ip
```

active_firewall.py - dodanie reguły iptables

```
def addRule(allert, sourceIp, rule):
        print('Adding rule: ' + rule)
        os.system(f"/sbin/iptables {rule}")
```

Dziękujemy za uwagę