

# Härtung Firewall

## Regelsätze

### Einrichten einer Firewall-Regel

**Aufgabe:** Stellen Sie sicher, dass SSH-Zugriffe (Port 22) auf firewall und webserver nur vom admin-Rechner aus angenommen werden.

- Wie lautet die entsprechende Firewall-Regel?
- Recherchieren Sie die entsprechenden Optionen!
- Prüfen Sie das Ergebnis!
- Wie unterscheiden sich die Aktionen DROP und REJECT?

## Firewall-Commands (auf Webserver und Firewall)

```
# Akzeptiert Verbindung von Adminserver
iptables -A INPUT -p tcp -s 10.10.10.4 --dport 22 -j ACCEPT

# Lässt keine erstmal keine Verbindung auf Port 22 zu
iptables -A INPUT -p tcp -s 0.0.0.0/0 --dport 22 -j DROP

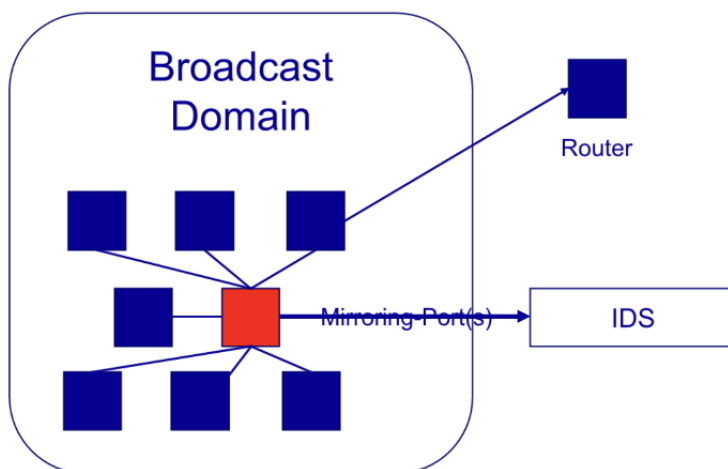
# Für jeden weiteren Eintrag - Input auf 2 damit die 0.0.0.0/0 Regel an Pos 1 bleibt
iptables -I INPUT 2 -p tcp -s hier neue ip --dport 22 -j ACCEPT
```

## Intrusion Detection System

## Was ist ein IDS (Intrusion Detection System)?

- Grundgedanke: Erkennen von ungewollten Verbindungsversuchen
- Zugriffe auf „nicht für die Öffentlichkeit“ vorgesehen Dienste erkennen
  - Einstufung als Angriffsversuch
  - Kann bereits über die Logging-Funktion einer Firewall erfolgen
- Fortgeschrittene Funktion:
  - Auswertung der Datenverkehre auf verdächtige Muster (“Indicators of Compromise“)
  - Verdächtige DNS-Anfragen
  - Verdächtige Inhalte in http-Anfragen
- Herausforderung: Hohe Anzahl von Fehlalarmen
  - Alert Fatigue – man sieht den Wald vor lauter Bäumen nicht
- Beispielprodukte (Open Source): Zeek (ehem. Bro), Suricata

## Aufbau eines IDS für das Monitoring eines gesamten Netzes



### Integration eines IDS ins Netz

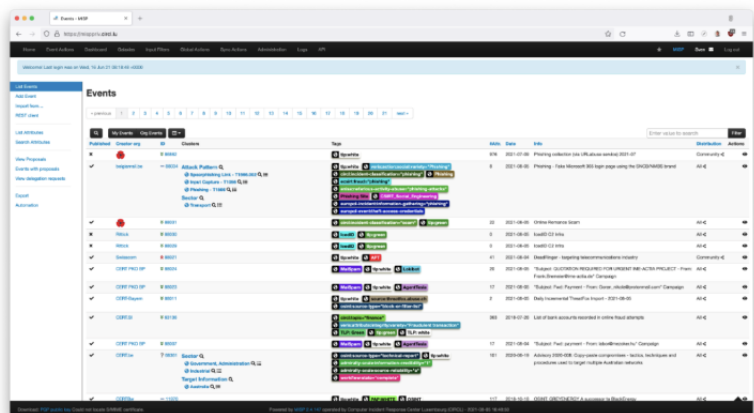
- Über einen speziellen Port am Switch wird der gesamte Datenverkehr der Broadcast-Domain an das IDS ausgeleitet
- Datenverkehr ist nur in eine Richtung – ggfs. können Adern aus dem Netzwirkabel entfernt werden
- Das IDS rekonstruiert den Datenverkehr und sucht nach verdächtigen Mustern
- Erfordert leistungsstarken Switch und leistungsstarken Rechner

### Aufbau eines Regelsatzes für ein IDS (Beispiel Suricata)

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ET MALWARE Win32/TrojanDownloader.Agent.BXA CnC Activity";  
flow:established,to_server; threshold:type limit, track by_src, seconds 600, count 1; content:"GET"; http_method;  
content:"Xingapp|2f 35 2e 30 20 28|windowsxue|29|"; depth:24; isdataat:!1,relative; http_user_agent;  
reference:md5,d4a8b93cb872a2817a1e7467ea449363; classtype:trojan-activity; sid:2033383; rev:1; metadata:affected_product  
Windows_XP_Vista_7_8_10_Server_32_64_Bit, attack_target Client_Endpoint, created_at 2021_07_21, deployment Perimeter,  
former_category ADWARE_PUP, performance_impact Low, signature_severity Minor, updated_at 2021_07_21;)  
  
alert tls $HOME_NET any -> $EXTERNAL_NET any (msg:"ET MALWARE Observed ZLoader CnC Domain in SNI"; flow:to_server,established;  
tls_sni; content:"yuidskadjna.com"; isdataat:!1,relative; classtype:trojan-activity; sid:2033422; rev:1;  
metadata:attack_target Client_Endpoint, created_at 2021_07_26, deployment Perimeter, former_category MALWARE,  
signature_severity Major, tag SSL_TLS_SNI, updated_at 2021_07_26, mitre_tactic_id TA0011, mitre_tactic_name  
Command_And_Control, mitre_technique_id T1573, mitre_technique_name Encrypted_Channel;)  
  
alert dns $HOME_NET any -> any any (msg:"ET MALWARE Socelars Related Domain in DNS Lookup"; dns_query;  
content:"www.cncode.pw"; nocase; depth:13; isdataat:!1,relative; reference:md5,f6c01214414fe2cedaa217c69ab093e1;  
classtype:bad-unknown; sid:2033607; rev:1; metadata:attack_target Client_Endpoint, created_at 2021_07_28, deployment Perimeter,  
former_category ADWARE_PUP, updated_at 2021_07_28, mitre_tactic_id TA0009, mitre_tactic_name Collection, mitre_technique_id  
T1005, mitre_technique_name Data_from_local_system;)
```

### Quellen für IDS-Regeln und Indicators of Compromise

- Eigene Analyse auf Grundlage erkannter Angriffe und Vorfälle
- Bezug von Listen von IDS-Herstellern
- Offene Listen aus Open Source-Projekten
- Threat Sharing-Plattformen und Communities (bspw. MISP vom CERT in Luxemburg)
- Trend: Vernetzung der Sicherheits-Teams und gegenseitiger Austausch von „Indicators of Compromise“



## Installation Suricata auf der Firewall

- Einrichten eines zusätzlichen, öffentlichen Netzwerkinterfaces vom Typ NAT (damit Zugriff auf das Internet möglich ist)
- Testen der Internet-Verbindung
- Installation Suricata:

```
apt install suricata suricata-update
```

- Editieren der Datei `/etc/suricata/suricata.yaml`, dort folgende Konfiguration eintragen:

```
address-groups:  
  HOME_NET: "[10.10.10.0/24]"
```

## ICMP-Alerts

### Test Suricata mit erster Regel

- Unsere erste Regel (Demo-Zwecke): Aufnahme einer Zeile in `/etc/suricata/suricata.yml`

```
rule-files:  
  - ba-rm.rules
```

- Erstellen der Datei `/etc/suricata/rules/ba-rm.rules` mit dem Inhalt:

```
alert icmp any any -> any any (msg: "ICMP Packet found";)
```

- Starten von Suricata mit:

```
suricata -i enp0s3 -i enp0s8 -D
```

- Suricata läuft jetzt im Hintergrund, beenden mit `killall suricata`
- Testen – Alarme finden sich unter `/var/log/suricata/fast.log`
- Ansehen mit `tail -f /var/log/suricata/fast.log` – Abbruch mit `ctrl-C`

```
# Start von Suricata  
# -D = Detached  
  
# suricata -i [INTERFACE1] -i [INTERFACE2] -D
```

```
suricata -i ens160 -i ens256 -D
```

## Commands zum Beenden eines Prozesses

```
# Prozess anzeigen  
ps aux | grep suricata  
  
# Prozess killen  
kill -9 [PID]  
  
# oder  
kill [PID]
```

## DNS-Alerts

### Netzwerksicherheit DNS-Alerts

- Aufnahme einer neuen Regel:

```
alert dns any any -> any any (msg: "DNS LOOKUP for Google"; dns_query;  
content:"google"; sid:1000001;)
```

- Testen!

Diese Regel in Suricata überwacht den DNS-Verkehr (Domain Name System) und löst eine Warnung aus, wenn eine DNS-Anfrage den Begriff "google" enthält.

- **alert dns:** Dies gibt an, dass die Regel auf DNS-Verkehr angewendet wird.
- **dns any any -> any any:** Dies bedeutet, dass die Regel für jede DNS-Anfrage von jeder Quelle zu jedem Ziel gilt.
- **(msg: "DNS LOOKUP for Google";** Dies ist die Nachricht, die in der Warnung angezeigt wird.
- **dns\_query;** Dies gibt an, dass die Regel nur auf DNS-Anfragen angewendet werden soll (keine Antworten).

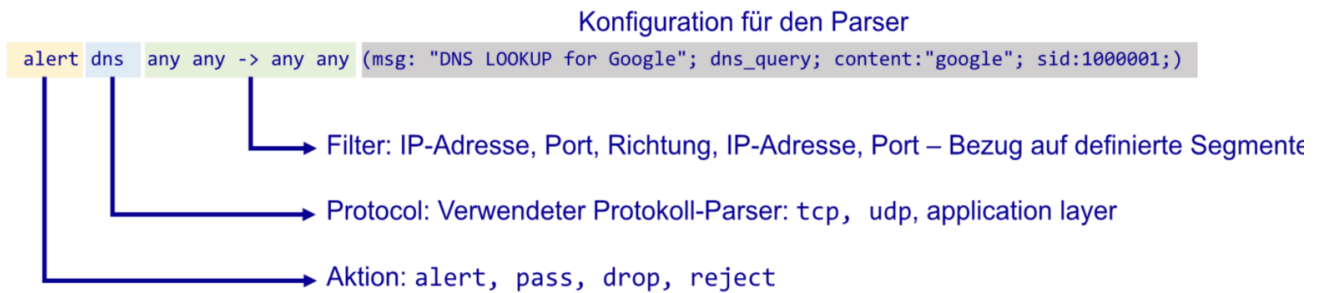
- **content: "google";** Dies ist die Bedingung, die erfüllt sein muss, damit die Warnung ausgelöst wird. Die DNS-Anfrage muss das Wort "google" enthalten.
- **sid:1000001;** Dies ist eine eindeutige ID für die Regel.

Mit folgenden Befehlen kann man testen:

```
# Google pingen
ping google.com

# DNS-Auflösung
nslookup google.com
```

## Netzwerksicherheit Anatomie einer Regel



- Aufgabe: Bauen einer Regel, die bei ausgehenden SSH-Verbindungen alarmiert.
- Dokumentation: <https://suricata.readthedocs.io/en/suricata-6.0.1/rules/index.html>
- Verwendung von Thresholds (Schwellen)

Vorlesung IT-Sicherheit, Dr. Sven Niedner

14

## Ausgehende SSH-Verbindungen erkennen und alerten

```
alert tcp 10.10.10.0/24 any -> any 22 (msg:"Ausgehende SSH-Verbindung erkannt"; flow:established,to_client;
sid:1000002; rev:1;)
```

## Suricata als Service in Debian machen

create user for **suricata**

```
useradd -r -s /usr/sbin/nologin suricata
```

change the **IFACE** at `/etc/default/suricata` and make it listen to our **interface**

```

GNU nano 2.9.3 /etc/default/suricata Modified
# Default config for Suricata

# set to yes to start the server in the init.d script
RUN=no

# Configuration file to load
SURCONF=/etc/suricata/suricata.yaml

# Listen mode: pcap, nfqueue or af-packet
# depending on this value, only one of the two following options
# will be used (af-packet uses neither).
# Please note that IPS mode is only available when using nfqueue
LISTENMODE=nfqueue

# Interface to listen on (for pcap mode)
IFACE=enp0s8,

# Queue number to listen on (for nfqueue mode)
NFQUEUE=0

# Load Google TCMALLOC if libtcmalloc-minimal4 is installed
# This _might_ give you very very small performance gain...
TCMALLOC="YES"

# Pid file
PIDFILE=/var/run/suricata.pid

```

change the owner of **/var/log/suricata** using command below

```
chown -R suricata:suricata /var/log/suricata
```

edit **interfaces** at **/etc/suricata/suricata.yaml** and make it listen to our **interface**

```

GNU nano 2.9.3 /etc/suricata/suricata.yaml
# This value is overridden by the SC_LOG_OP_FILTER env var.
default-output-filter:

# Define your logging outputs. If none are defined, or they are all
# disabled you will get the default: console output.
outputs:
- console:
    enabled: yes
    # type: json
- file:
    enabled: yes
    level: info
    filename: suricata.log
    # type: json
- syslog:
    enabled: no
    facility: local5
    format: "[%i] <%d> -- "
    # type: json

##
## Step 3: Configure common capture settings
##
## See "Advanced Capture Options" below for more options, including Netmap
## and PF_RING.
##

# Linux high speed capture support
af-packet:
- interface: enp0s8
  # Number of receive threads. "auto" uses the number of cores
  #threads: auto
  # Default clusterid. AF_PACKET will load balance packets based on flow.
  cluster-id: 99
  # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
  # This is only supported for Linux kernel > 3.1
  # possible value are:
  # * cluster_flow: all packets of a given flow are sent to the same socket
  # * cluster_cpu: all packets treated in kernel by a CPU are sent to the same socket
  # * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
  # socket. Requires at least Linux 3.14.
  # * cluster_ebpf: eBPF file load balancing. See doc/userguide/capture-hardware/ebpf-xdp.rst for
  # more info.
  # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
  # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
  cluster-type: cluster_flow
  # In some fragmentation cases, the hash can not be computed. If "defrag" is set
  # to yes, the kernel will do the needed defragmentation before sending the packets.
  defrag: yes

```

start the **Suricata** and make sure it has been running well

```
systemctl start suricata
systemctl status suricata
```

---

Revision #14

Created 16 August 2024 08:07:22 by Max

Updated 6 September 2024 08:50:29 by Max