=== ClamAV Realtime Scanner mit IPTables Blockade und Email-Benachrichtigung ===

Hier ist eine komplette Lösung in Python, die Dateien in einem Verzeichnis überwacht, bei Virenfunden den ausgehenden Verkehr blockiert und eine Email sendet. Die Lösung besteht aus einem Hauptskript (als Systemdienst) und einem Nebenskript zur Aufhebung der Blockade.

=== Funktionsbeschreibung ===

1. Hauptskript (clamav\_inotify\_service.py):
   * Überwacht ein Verzeichnis mit inotify
   * Scannt neue/changed Dateien mit ClamAV
   * Bei Virenfund:
     + Blockiert ausgehenden Traffic mit iptables
     + Sendet Email via Exchange
   * Läuft als Systemdienst im Hintergrund
2. Nebenskript (unblock\_reset\_service.py):
   * Hebt die IPTables-Blockade auf
   * Startet den Hauptdienst neu
   * Wird manuell nach Bereinigung aufgerufen

=== Voraussetzungen ===

Benötigte Pakete installieren:

bash

sudo apt-get install python3 python3-pip clamav clamav-daemon inotify-tools

sudo pip3 install pyclamd python-iptables exchangelib

=== Hauptskript (clamav\_inotify\_service.py) ===

python

*#!/usr/bin/env python3*

import os

import sys

import pyclamd

import time

import smtplib

from email.mime.text import MIMEText

import iptc

import logging

from systemd.journal import JournalHandler

from exchangelib import Credentials, Account, Message, Mailbox

*# === Konfiguration ===*

WATCH\_DIR = "/path/to/watch" *# Zu überwachendes Verzeichnis*

EMAIL\_RECIPIENT = "admin@example.com" *# Empfänger-Email*

EMAIL\_USER = "your\_email@exchange.com" *# Exchange Email*

EMAIL\_PASSWORD = "your\_password" *# Exchange Passwort*

EXCHANGE\_SERVER = "outlook.office365.com" *# Exchange Server*

*# === Logging einrichten ===*

log = logging.getLogger('clamav\_inotify\_service')

log.addHandler(JournalHandler())

log.setLevel(logging.INFO)

def send\_email(virus\_name, file\_path):

"""Sendet Email über Exchange bei Virenfund"""

try:

credentials = Credentials(username=EMAIL\_USER, password=EMAIL\_PASSWORD)

account = Account(primary\_smtp\_address=EMAIL\_USER,

credentials=credentials,

autodiscover=True,

access\_type='delegate')

msg = Message(

account=account,

subject=f"Virus gefunden: {virus\_name}",

body=f"Datei {file\_path} enthält Virus {virus\_name}. Ausgehender Verkehr wurde blockiert.",

to\_recipients=[Mailbox(email\_address=EMAIL\_RECIPIENT)]

)

msg.send()

log.info(f"Email an {EMAIL\_RECIPIENT} gesendet")

except Exception as e:

log.error(f"Email konnte nicht gesendet werden: {str(e)}")

def block\_outgoing\_traffic():

"""Blockiert ausgehenden Traffic mit iptables"""

try:

chain = iptc.Chain(iptc.Table(iptc.Table.FILTER), "OUTPUT")

rule = iptc.Rule()

rule.target = iptc.Target(rule, "DROP")

chain.insert\_rule(rule)

log.info("Ausgehender Traffic wurde blockiert")

except Exception as e:

log.error(f"Fehler beim Blockieren des Traffics: {str(e)}")

def scan\_file(file\_path):

"""Scannt eine Datei mit ClamAV"""

try:

cd = pyclamd.ClamdAgnostic()

scan\_result = cd.scan\_file(file\_path)

if scan\_result is not None:

virus\_name = scan\_result[file\_path][1]

log.warning(f"Virus gefunden: {virus\_name} in {file\_path}")

*# Blockiere Traffic und sende Email*

block\_outgoing\_traffic()

send\_email(virus\_name, file\_path)

*# Beende Dienst nach Virenfund*

sys.exit(1)

except Exception as e:

log.error(f"Fehler beim Scannen: {str(e)}")

def monitor\_directory():

"""Überwacht Verzeichnis mit inotify"""

import inotify.adapters

notifier = inotify.adapters.Inotify()

notifier.add\_watch(WATCH\_DIR)

log.info(f"Beginne mit der Überwachung von {WATCH\_DIR}")

for event in notifier.event\_gen():

if event is not None:

(header, type\_names, watch\_path, filename) = event

file\_path = os.path.join(watch\_path, filename)

*# Nur auf bestimmte Events reagieren*

if 'IN\_CLOSE\_WRITE' in type\_names or 'IN\_MOVED\_TO' in type\_names:

if os.path.isfile(file\_path):

log.info(f"Neue/changed Datei: {file\_path}")

scan\_file(file\_path)

def main():

log.info("ClamAV Inotify Service gestartet")

try:

monitor\_directory()

except KeyboardInterrupt:

log.info("Service wird beendet")

except Exception as e:

log.error(f"Fehler im Hauptprozess: {str(e)}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

=== Nebenskript (unblock\_reset\_service.py) ===

python

*#!/usr/bin/env python3*

import iptc

import subprocess

import logging

from systemd.journal import JournalHandler

import os

*# === Logging einrichten ===*

log = logging.getLogger('unblock\_reset\_service')

log.addHandler(JournalHandler())

log.setLevel(logging.INFO)

def unblock\_outgoing\_traffic():

"""Hebt die IPTables-Blockade auf"""

try:

table = iptc.Table(iptc.Table.FILTER)

chain = iptc.Chain(table, "OUTPUT")

*# Lösche alle DROP-Regeln in OUTPUT*

for rule in chain.rules:

if rule.target.name == "DROP":

chain.delete\_rule(rule)

log.info("Ausgehende Traffic-Blockade aufgehoben")

except Exception as e:

log.error(f"Fehler beim Aufheben der Blockade: {str(e)}")

def restart\_main\_service():

"""Startet den Hauptdienst neu"""

try:

subprocess.run(["systemctl", "restart", "clamav\_inotify.service"], check=True)

log.info("Hauptdienst wurde neu gestartet")

except subprocess.CalledProcessError as e:

log.error(f"Fehler beim Neustart des Hauptdienstes: {str(e)}")

def main():

log.info("Starte Aufhebung der Blockade und Neustart")

unblock\_outgoing\_traffic()

restart\_main\_service()

log.info("Prozess abgeschlossen")

if \_\_name\_\_ == "\_\_main\_\_":

main()

=== Systemd Service Einrichtung ===

1. Erstellen Sie die Service-Datei für den Hauptdienst:

bash

sudo tee /etc/systemd/system/clamav\_inotify.service <<EOF

[Unit]

Description=ClamAV Realtime Scanner with IPTables Blocking

After=network.target clamav-daemon.service

[Service]

Type=simple

ExecStart=/usr/bin/python3 /path/to/clamav\_inotify\_service.py

Restart=on-failure

RestartSec=5s

[Install]

WantedBy=multi-user.target

EOF

1. Aktivieren und starten Sie den Dienst:

bash

sudo systemctl daemon-reload

sudo systemctl enable clamav\_inotify.service

sudo systemctl start clamav\_inotify.service

=== Verwendung ===

1. Der Hauptdienst läuft automatisch und überwacht das konfigurierte Verzeichnis
2. Bei Virenfund wird:
   * Ausgehender Traffic blockiert
   * Email gesendet
   * Dienst gestoppt
3. Nach Bereinigung:

bash

python3 /path/to/unblock\_reset\_service.py

Dies hebt die Blockade auf und startet den Hauptdienst neu

=== Wichtige Hinweise ===

* Testen Sie das Skript in einer kontrollierten Umgebung
* Passen Sie die Email-Konfiguration an Ihren Exchange-Server an
* Stellen Sie sicher, dass der Benutzer, der den Dienst ausführt, die nötigen Rechte hat
* Überwachen Sie die Logs mit: journalctl -u clamav\_inotify.service -f

Die Skripte sind modular aufgebaut und können leicht an spezifische Anforderungen angepasst werden.