Bevezető

- https://docs.clamav.net/manual/Usage/Scanning.html
- https://linuxcapable.com/install-clamav-on-fedora-linux/#:~:text=To%20customize%20ClamAV%20settings%2C%20such,conf%20.

A ClamAV egy nyílt forráskódú (GPLv2) vírusirtó eszközkészlet, amelyet különösen e-mail átjárók e-mailjeinek vizsgálatára terveztek. Számos segédprogramot biztosít, beleértve egy rugalmas és skálázható több szálú démont, egy parancssori szkennert és egy fejlett eszközt az automatikus adatbázis-frissítésekhez.

> **Tipp:** A ClamAV nem egy hagyományos vírusirtó vagy végpontbiztonsági csomag. Egy teljes funkcionalitású modern végpontbiztonsági csomaghoz nézze meg a Cisco Secure Endpoint-et. További részletekért lásd az "kapcsolódó termékek" részt alább.

A ClamAV-ot a Cisco Systems, Inc. hozta létre.

A ClamAV-nak sokszínű ökoszisztémája van közösségi projektekből, termékekből és egyéb eszközökből, amelyek vagy a ClamAV-ra támaszkodnak a rosszindulatú programok észlelési képességeinek biztosításához, vagy kiegészítik a ClamAV-ot új funkciókkal, mint például a harmadik féltől származó aláírási adatbázisok jobb támogatása, grafikus felhasználói felületek (GUI) és még sok más.

A ClamAV gyors fájlvizsgálatra lett tervezve.
Valós idejű védelem (csak Linux). A ClamOnAcc kliens a ClamD szkennelő démonhoz valós idejű szkennelést biztosít a modern Linux verziókon. Ez magában foglalja az opcionális képességet, hogy a fájlhozzáférést blokkolja, amíg a fájl nem lett átvizsgálva (valós idejű megelőzés).
A ClamAV milliónyi vírust, férget, trójait és egyéb rosszindulatú programot észlel, beleértve a Microsoft Office makró vírusokat, mobil rosszindulatú programokat és egyéb fenyegetéseket.

A ClamAV bytecode aláírási futtatókörnyezete, amelyet vagy az LLVM, vagy a saját bytecode értelmezőnk hajt végre, lehetővé teszi a ClamAV aláírásírók számára, hogy nagyon összetett észlelési rutinokat hozzanak létre és terjesszenek, valamint távolról javítsák a szkenner funkcionalitását.

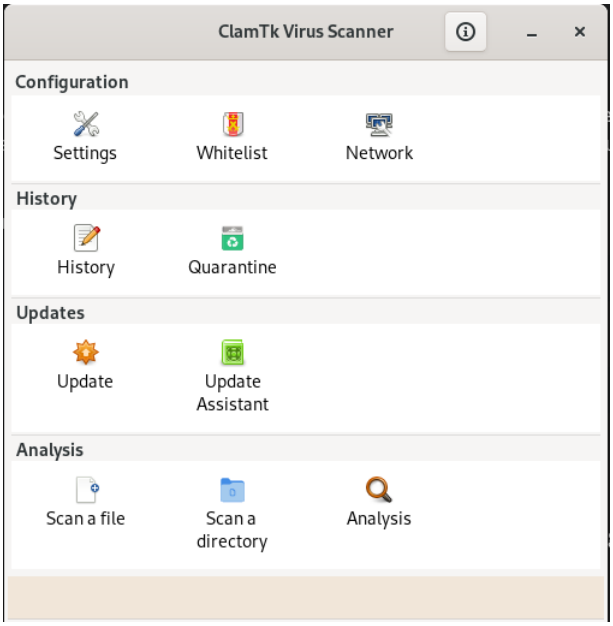
Installálás

Alap szoftver installáció:

```
$ sudo dnf install clamav clamd clamav-update
```

GUI:

```
$ sudo dnf install clamtk
```



Mappa rekurzív sacnnaelése:

```
clamscan -r otpbank/
...
----------- SCAN SUMMARY -----------
Known viruses: 8704956
Engine version: 1.0.8
Scanned directories: 6
Scanned files: 100
Infected files: 0
Data scanned: 0.04 MB
Data read: 0.04 MB (ratio 1.00:1)
Time: 9.100 sec (0 m 9 s)
Start Date: 2025:03:10 18:23:06
End Date:   2025:03:10 18:23:16
```

Automatikus vírus db frissítés

A freshclam démonként fut (freshclam daemon):

- csatlakozik a ClamAV központi adatbázis szervereihez,
- letölti az új main.cvd, daily.cvd, bytecode.cvd adatbázisfájlokat

Első manuális frissítés:

Állítsuk le az elindított freschclam programot, hogy manuálisan el tudjuk végezni az első frissítést:

```
$ sudo systemctl stop clamav-freshclam
```

Manuálisan frissítjük a vírus adatbázist:

```
$ sudo freshclam
```

Beállítjuk, hogy mindig fusson:

```
$ sudo systemctl enable --now clamav-freshclam
```

Frissítési gyakoriság beállítása

- Config fájl: **/etc/freshclam.conf**
- Logok: /var/log/clamav/freshclam.log

```
# Number of database checks per day.
# Default: 12 (every two hours)
#Checks 24
```

ClamAV demon beállítása

Config fájl: /etc/clamd.d/scan.conf

A clamd démon futtatása attól függ, hogy milyen módon szeretnéd használni a ClamAV-t:

1. Ha csak időnként szeretnéd futtatni a víruskeresést (pl. manuálisan vagy cron job segítségével), akkor a clamd démon nem szükséges folyamatosan futnia. Ehelyett a clamscan parancsot használhatod egy adott könyvtár átvizsgálására.
2. Ha valós idejű vagy rendszeres automatikus víruskeresést szeretnél (pl. egy fájlszerver védelme érdekében), akkor érdemes elindítani a clamd démont, mivel ez jelentősen felgyorsítja a vizsgálatokat. A clamd előre betölti a vírusadatbázist és gyorsabb elemzést tesz lehetővé, mint az egyenkénti clamscan futtatás.

Milyen user nevében fusson a clamd?

Tulajdonság	`clamscan` **(default)**	`saját user`	`root` **(admin jog)**
Hozzáférés saját fájljaidhoz	⚠ korlátozott	✔ teljes	✔ teljes
Hozzáférés rendszerfájlokhoz	✖	✖	✔
Védettség hibás kódfuttatás ellen	✔ erős izoláció	⚠ gyengébb izoláció	✖ nincs izoláció (magas kockázat)
Vírusirtó által lefedett terület	⚠ részleges	✔ saját home könyvtár teljesen	✔ globális lefedettség
Konfigurációs bonyolultság	✔ alapértelmezett	⚠ override szükséges	⚠ override + biztonsági kockázat

Miért nem rootként fut alapból a clamd?

A clamd általában clamav vagy clamscan nevű korlátozott rendszerfelhasználóként fut. Ennek oka:

- minimalizálja a kárt, ha valaki biztonsági hibát talál a clamd-ban,
- nem akarjuk, hogy egy hálózaton keresztül vezérelhető víruskereső motor root jogosultságú legyen.

✔ De mit nyersz azzal, ha mégis rootként futtatod?

Teljes fájlrendszer hozzáférés
pl. /root, /home/adam/.config, zárolt fájlok stb.
Kevesebb File path check failure hiba
Egyszerűbb beállítás – nem kell csoportokat, jogosultságokat hangolgatni

⚠ Mi a kockázat?

Egy távoli támadó, aki exploitál egy hibát a clamd processzben (pl. fertőzött fájl manipulált szkennelésével), root jogot szerezhet a rendszereden.
Ez különösen akkor veszélyes, ha a clamd TCP socketet is szolgáltat (pl. port 3310-on).

Saját user nevében?

✔ Mit nyersz azzal, ha clamd az saját user nevében fut?

- Hozzáférés a teljes /home/adam/ struktúrához (beleértve .config, .local, .mozilla, stb.)
- Nem kell chmod, chgrp, vagy extra csoport
- Nem rootként fut, tehát kisebb a támadási felület

⚠ Mit nem fog tudni így a clamd?

- Nem fér hozzá más felhasználók fájljaihoz
- Nem lát rendszerfájlokat (pl. /etc, /bin, /var)
- Nem tudja vizsgálni pl. /root, vagy máshol levő fájlokat, amikhez root kellene

clamscan user nevében

✔ Előnyök:

- Biztonságos alapértelmezett beállítás
  - A clamscan felhasználó jogai szigorúan korlátozottak, nincs írás- vagy olvasási jog a legtöbb helyre.
  - Ha valaki kihasznál egy clamd-ban található sebezhetőséget, akkor csak clamscan jogosultsággal fér hozzá a rendszerhez.
- Kompatibilis a disztribúció beállításaival
  - A Fedora (és más rendszerek) clamd@scan szolgáltatása alapból ehhez van konfigurálva.
  - A /run/clamd.scan/clamd.sock socket fájlt clamscan:clamscan user/csoport hozza létre.
- Támogatott, stabil modell: A ClamAV fejlesztői így tervezték, így ez a legjobban tesztelt mód.

🟡 Hátrányok:

- Korlátozott hozzáférés a fájlrendszerhez
  - A clamscan user nem fér hozzá a legtöbb felhasználói home könyvtárhoz, különösen a zárt mappákhoz (mint .config, .cache, stb.)
- További konfiguráció szükséges, ha szélesebb körű fájlhozzáférést akarsz:
  - Csoporttagságokat, fájlrendszer-jogosultságokat kell módosítanod (pl. virusgroup, chmod, chgrp)
  - Ez viszont könnyen biztonsági rést nyithat, ha nem figyelsz oda
- Nem fér hozzá root-only fájlokhoz: Pl. /root, /etc/shadow, vagy más felhasználók fájljai

User beállítása

Az alábbi két lehetőségből válasszunk egyet.

Futtatás clamscan userrel

Magyarázat:

Ez a default működés, ebben az esetben a clamad csak azokat a fájlokat fogja tudni átvizsgálni, ahol a csoport tagok is olvashatják a fájlt. A csak tulajdonos számára hozzáférhető fájlokra a clamd hibát fog dobni.

> **WARNING**: multi user környezetben is működik, de akkor a clamscan felhasználót minden user csoportjához hozzá kell adni.

Nem fog hozzáférni semmi olyanhoz, amit a csoport nem olvashat:

```
File path check failure on: /home/adam/.config/google-chrome/Default/Cookies-journal
```

Ennek a fájlnak ezek a beállításai:

```
$ ls -l /home/adam/.config/google-chrome/Default/Cookies-journal
-rw-------. 1 adam adam 0 Mar 21 16:39 /home/adam/.config/google-chrome/Default/Cookies-journal
```

Csoportok létrehozása:

```
$ sudo groupadd virusgroup
```

A daemon a clamscan felhasználó nevében fog futni, ezért hozzá kell adni a csoporthoz.

```
$ sudo usermod -aG virusgroup clamscan
```

Saját felhasználó hozzáadása a virusgroup-hoz, hogy a daemon elérhesse a saját fájljainkat?

```
$ sudo usermod -aG virusgroup $USER
```

És a clamscan-t hozzáadjuk a saját csoportunkhoz is, hogy láthassa azokat a fájlokat amik az 'adam' csoport olvashat:

```
$ sudo usermod -aG adam clamscan
```

Tagok listázása:

```
$ getent group virusgroup
virusgroup:x:966:clamupdate,clamscan,adam
```

Futtatás saját user nevében

Ez a megoldás multi user környezetben nem fog működni, de ha csak egy user van egy gépen (tipikusan igen), akkor ez egy optimális megoldás.

```
$ sudo EDITOR=mcedit systemctl edit clamd@scan

Successfully installed edited file '/etc/systemd/system/clamd@scan.service.d/override.conf'.
```

Oda ahol mondja, hogy a két comment közé írjunk, ezt kell írni:

```
[Service]
User=adam
Group=adam
RuntimeDirectory=clamd.scan
RuntimeDirectoryMode=0770
```

(Az utolsó két sorban nem vagyok biztos)

A log fájlt adjuk az adam tulajdonába:

```
sudo touch /var/log/clamd.scan
sudo chown adam:adam /var/log/clamd.scan
sudo chmod 640 /var/log/clamd.scan
```

ClamAV daemon újraindítása:

```
sudo systemctl restart clamd@scan
```

Ellenőrizzük kinek a nevében fut:

```
$ ps -C clamd -o pid,euser,egroup,cmd
    PID EUSER    EGROUP   CMD
 206815 adam     adam     /usr/sbin/clamd -c /etc/clamd.d/scan.conf
```

Általános beállítások:

```
LogFile /var/log/clamd.scan
LogVerbose yes
LogRotate yes

User adam

# Default: 100M
#MaxFileSize 400M
```

Automatikus indítás

Tegyük auto start-ra a clamd-t:

```
$ sudo systemctl enable --now clamd@scan
```

Indítás és teszt

ClamAV daemon újraindítása:

```
sudo systemctl restart clamd@scan
```

Ellenőrzés:

```
$ sudo systemctl status clamd@scan
● clamd@scan.service - clamd scanner (scan) daemon
     Loaded: loaded (/usr/lib/systemd/system/clamd@.service; enabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/service.d
             └─10-timeout-abort.conf, 50-keep-warm.conf
     Active: active (running) since Wed 2025-03-12 18:23:27 CET; 37s ago
```

Test scannelés:

```
$ clamscan index.md
Loading:     8s, ETA:   0s [========================>]    8.71M/8.71M sigs
Compiling:   1s, ETA:   0s [========================>]       41/41 tasks

/home/adam/repositories/Other/berkiadam-github/wiki/linux/clamav/index.md: OK

----------- SCAN SUMMARY -----------
Known viruses: 8706011
Engine version: 1.0.8
Scanned directories: 0
Scanned files: 1
Infected files: 0
Data scanned: 0.01 MB
Data read: 0.00 MB (ratio 2.00:1)
Time: 9.489 sec (0 m 9 s)
Start Date: 2025:03:21 14:03:04
End Date:   2025:03:21 14:03:13
```

Troubleshooting:

```
$ journalctl -xeu clamd@scan
```

Időzített rendszeres ellenőrzés futtatása

TODO...

SELinux

Ha a SELinux be van kapcsolva, akkor fontos, hogy beállítsuk a virus kereső működést.

You must tell SELinux about this by enabling the 'antivirus_can_scan_system' boolean:

```
setsebool -P antivirus_can_scan_system 1
sudo usermod -aG virusgroup adam
```

On-Access scanning

- https://docs.clamav.net/manual/Usage/Scanning.html#on-access-scanning
- https://docs.clamav.net/manual/OnAccess.html

> **WARNING**: On-Access requires a kernel version >= 3. This is because it leverages a kernel api called fanotify to block processes from attempting to access malicious files. This prevention occurs in kernel-space, and thus offers stronger protection than a purely user-space solution.

clamonacc program segítségével lehet on-access scanning-et futtatni.

Socket beállítások

A clamonacc egy lokális socke-ten küldi át majd a scannelendő fájlokat a clmad-nek. Ehhez létre kell hozni kézzel egy socket fájlt és be kell állítsuk rajta a megfelelő jogosultságokat és user-eket.

Reláció a clamonacc és a clamd között:

```
clamonacc (root)  ⟷  fanotify esemény
        |
        ↓
clamd (clamav user) ⟷ fájlmegnyitás, szkennelés
```

Komponens	Jogosultság	Feladat
`clamonacc`	`root`	eseményfigyelés (`fanotify`)
`clamd`	`clamav` vagy `clamscan`	fájlok megnyitása és vírusvizsgálata

Socket létrehozása az on-access vizsgálathoz:

```
sudo mkdir -p /run/clamd.scan
sudo chown adam:virusgroup /run/clamd.scan
```

```
sudo chmod 770 /run/clamd.scan
```

> **WARNING**: Itt fontos, hogy annak a felhasználónak a tulajdonába adjuk a socket-et akinek a nevében futtatjuk a clamav-t. Jelen esetben a saját user-ünk

> **NOTE**: Ez egy átmeneti mappa, minden induláskor a systemctl újra létre fogja hozni azokkal a beállításokkal ami a configban van. Tehát ha ott rossz user és group van, akkor újraindítás után már az ő tulajdonában lesz és megint csak nem fog tudni olvasni belőle a camad.

Socket beállítása a **/etc/clamd.d/scan.conf** fájlban:

```
LocalSocket /run/clamd.scan/clamd.sock
LocalSocketMode 660
```

Indítsuk újra a clmad-t és nézzük meg hogy hallgatózik e a socket-en:

```
sudo systemctl restart clamd@scan
```

Nézzük meg hogy figyel e a socket-en:

```
$ ss -lx | grep clamd
u_str LISTEN 0      200       /run/clamd.scan/clamd.sock 24800             * 0
```

Clamonacc indító service módosítása

A clamonacc gyári indításából hiányzik a --fdpass, ami szükséges, ha clamonacc nem clamd nevében fut.

```
sudo EDITOR=mcedit systemctl edit clamav-clamonacc.service
```

Úgy, hogy a # jelek között ne legyen új sor:

```
### Anything between ...
[Service]
ExecStart=
ExecStart=/usr/sbin/clamonacc -F --fdpass --config-file=/etc/clamd.d/scan.conf
### Edit below
```

Az első ExecStart= sor kinullázza az eredetit, így nem lesz duplikált.

Mentés után ezt látjuk:

```
Successfully installed edited file '/etc/systemd/system/clamav-clamonacc.service.d/override.conf'.
```

Töltsük újra a service konfigurációt:

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
```

scan.conf változtatások

on-Access beállítása a **/etc/clamd.d/scan.conf** fájlban:

```
OnAccessMaxFileSize 10M
OnAccessIncludePath /home
OnAccessPrevention yes

# Ezekre ne vizsgáljon
OnAccessExcludePath /home/adam/.config
```

A root és a clamd felhasználókat ki kell zárni a scannelésből. Ez azért fontos, mert ha a clamd saját maga által írt fájlokat is ellenőrzi, végtelen ciklusba kerülhet.

```
# Kizárja a root UID-ját a szkennelésből (ajánlott)
OnAccessExcludeRootUID yes

# Kizárja a `clamd` felhasználót a szkennelésből
OnAccessExcludeUname clamscan
```

ClamAV daemon újraindítása:

```
sudo systemctl restart clamd@scan
```

Clamonacc újraindítása:

```
sudo systemctl restart clamonacc
```

Ellenőrzés:

```
# sudo systemctl status clamav-clamonacc.service
● clamav-clamonacc.service - ClamAV On-Access Scanner
     Loaded: loaded (/usr/lib/systemd/system/clamav-clamonacc.service; disabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/service.d
             └─10-timeout-abort.conf, 50-keep-warm.conf
             /etc/systemd/system/clamav-clamonacc.service.d
             └─override.conf
     Active: active (running) since Mon 2025-03-24 21:11:52 CET; 2min 2s ago
 Invocation: 8448a0a166c449d8a2c4dce61add2da8
       Docs: man:clamonacc(8)
             man:clamd.conf(5)
             https://docs.clamav.net/
   Main PID: 10435 (clamonacc)
      Tasks: 8 (limit: 76567)
     Memory: 92.9M (peak: 94.2M)
        CPU: 2.433s
     CGroup: /system.slice/clamav-clamonacc.service
             └─10435 /usr/sbin/clamonacc -F --fdpass --config-file=/etc/clamd.d/scan.conf

Mar 24 21:11:52 fedora systemd[1]: Started clamav-clamonacc.service - ClamAV On-Access Scanner.
Mar 24 21:11:54 fedora clamonacc[10435]: ClamInotif: watching '/home' (and all sub-directories)
Mar 24 21:11:54 fedora clamonacc[10435]: ClamInotif: excluding '/home/adam/.config' (and all sub-directories)
```

Láthatjuk, hogy

- jó helyről szedi fel a configot: /etc/clamd.d/scan.conf
- szerepel a **--fdpass** az indító parancsban
- a /home mappát figyeli
- a .config mappa kivétel

Fontos, hogy a clamonacc root-két fusson:

```
$ ps -C clamonacc -o pid,euser,egroup,cmd
    PID EUSER    EGROUP   CMD
 162360 root     root     /usr/sbin/clamonacc -F --config-file=/etc/clamd.d/scan.conf
```

Automatikus indítása a clamonacc-nek:

```
$ sudo systemctl enable --now clamonacc
```

System logban:

```
$ journalctl -f
...
...
Mar 24 15:16:16 fedora clamonacc[22116]: ClamInotif: watching '/home' (and all sub-directories)
Mar 24 15:16:16 fedora clamonacc[22116]: ClamInotif: excluding '/home/adam/.config' (and all sub-directories)
```

Socket Tesztelés

Nézzük meg folyik a kommunikáció a local socket-en, ehhez a **strace** programot fogjuk használni:

```
$ sudo dnf install strace
```

Majd hallgassunk bele. A localsocket-en bináris forgalom közlekedik, így csak azt fogjuk látni, hogy zajlik az élet, hogy milyen fájlokat küld át a clamonacc, azt nem :

```
$ sudo strace -p $(pidof clamd) -s 100 -e trace=read,write
strace: Process 1784 attached
read(6, "\0", 1025)                     = 1
read(6, "\0", 1025)                     = 1
read(6, "\0", 1025)                     = 1
read(6, "\0", 1025)                     = 1
read(6, "\0", 1025)                     = 1
read(6, "\0", 1025)                     = 1
....
```

Karantén használata

sudo clamonacc --move=/var/quarantine --log=/var/log/clamonacc.log --fdpass

---

Finomhangolás

Az **on-access scanning** borzasztó erőforrás igényes, ha nem korlátozzuk le, hogy mire futhat le annyira megfoghatja a CPU-t, hogy használhatatlan lesz a gép. Ezért fontos, hogy finomhangoljuk hogy hogyan és mire fusson az on-access scan hatására a clamd.

Milyen mappákat vizsgáljon

Az összes olyan mappát excludálni kell az on-access scannelés alól, ami:

- config fájlokat tartalmaz, tipikusan a . kezdetű mappák
- GIT repókat tartalmazó mappák
- programok bináris fájlrajit tartalmazó mappák, pl a vscode bináris mappáját ha figyeli, akkor kb 20mp lesz mire el tud indulni. Ezeket muszáj kivenni a vizsgálat alól.

```
OnAccessExcludePath /home/adam/.cache
OnAccessExcludePath /home/adam/.cassandra
OnAccessExcludePath /home/adam/.cert
...
OnAccessExcludePath /home/adam/repositories
...
```

Email-ek vizsgálata

Ha használunk email vastag klienst, pl Evolution, akkor figyelni kell rá, hogy az email kezelő email mappáját szintén figyelje az on access clamAV.

... TODO...

Config finomhangolása

- Miket excludáljunk
- Mit scanneljen és hogy?
    - Data Loss Prevention (DLP)
    - Mail files
    - scan Documents
    - Executable files
    - Heuristic Alerts
- #VirusEvent /opt/send_virus_alert_sms.sh

Értesítés virus eseményről

Alapértelmezetten, ha az on access scan hatására a clamAV vírust talál, akkor arról csak a system logból értesülhetünk. Viszont a ClamAV biztosít egy script futtatási lehetőséget virus eseménykor. Ebben a script-ben tudunk GNOME alertet küldeni, vagy akár emailt, amire szükségünk van, és ebben a script-ben tudjuk áthelyezni karanténba a fertőzött fájlt, mert alapértelmezetten a clamonacc csak blokkolni fogja a hozzáférést.

A fertőzött fájl neve és a vírus neve az alábbi két környezeti változóba kerül mindig beállításra, mielőtt a clamd meghívná a vírus esemény scriptet:

- $CLAM_VIRUSEVENT_FILENAME
- $CLAM_VIRUSEVENT_VIRUSNAME

Karantén létrehozása

```
sudo mkdir -p /var/quarantine
sudo chown adam:adam /var/quarantine
```

```
sudo chmod 700 /var/quarantine
```

Virus esemény script

Keressük meg a saját UserID-nakt, mert fontos, hogy annak a nevében fusson majd a script

```
$ id -u
1000
```

- Fontos, hogy használatban lévő fájlt nem tudunk karanténba helyezni, mert a teljes X-et be tudja fagyasztani, ezért meg kell vizsgálni, hogy nincs e használatban. Pl Ha meg akarom nyitni olvasásra, az használatnak számít, ilyenkor nem tudom mozgatni.
- Azt a fájlt amit nem fog senki, pl csak átmásolom egy mappából egy másikba, azt tudom karanténba rakni
- A megnyitás alatt lévő fáj tartalmát sem tudjuk módosítani, nem tudjuk beleírni, hogy virusos volt
- Lehetne azt csinálni, hogy kilőjük azt a programot, ami fogja a fájlt, de ez adatvesztéshez vezethet, ez túl veszélyes.

/usr/local/bin/clamav-alert.sh

```
#!/bin/bash

export DISPLAY=:0
export DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus

# --- Beállítások ---
QUARANTINE_DIR="/var/quarantine"
LOGFILE="/var/log/clamav-alert.log"

# --- Környezeti változók ---
FILENAME="${CLAM_VIRUSEVENT_FILENAME}"
VIRUSNAME="${CLAM_VIRUSEVENT_VIRUSNAME}"
TIMESTAMP=$(date '+%Y-%m-%d %H:%M:%S')

# --- Ellenőrzés ---
if [ -z "$FILENAME" ] || [ -z "$VIRUSNAME" ]; then
    echo "$TIMESTAMP [ERROR] Missing required environment variables." >> "$LOGFILE"
    exit 1
fi

# --- Ellenőrizzük, hogy a fájlt használja-e valami ---
if lsof "$FILENAME" > /dev/null 2>&1; then
    echo "$TIMESTAMP [WARN] File in use, skipping quarantine: $FILENAME" >> "$LOGFILE"
    notify-send -u normal -a ClamAV "⚠ Virus detected" "In use: $FILENAME\nVirus: $VIRUSNAME"
    exit 0
fi

# --- Mozgatás karanténba ---
BASENAME=$(basename "$FILENAME")
TARGET="$QUARANTINE_DIR/$BASENAME.$(date +%s)"

if mv "$FILENAME" "$TARGET"; then
    echo "$TIMESTAMP [INFO] Moved infected file to quarantine: $TARGET (virus: $VIRUSNAME)" >> "$LOGFILE"
    notify-send -u critical -a ClamAV "☣ Virus Found!" "File quarantined:\n$TARGET\nVirus: $VIRUSNAME"
else
    echo "$TIMESTAMP [ERROR] Failed to move file: $FILENAME" >> "$LOGFILE"
    notify-send -u critical -a ClamAV "☣ Virus Found!" "Failed to quarantine:\n$FILENAME\nVirus: $VIRUSNAME"
fi
```

Jogosultságok beállítása: annak a felhasználónak a birtokába kell adni, akinek a nevében fut a clamd, ami az esetükben a saját user-ünk:

```
sudo chown adam:adam /usr/local/bin/clamav-alert.sh
sudo chmod 750 /usr/local/bin/clamav-alert.sh
```

ClamD config módosítás

Állítsuk be a **/etc/clamd.d/scan.conf**-ba:

```
VirusEvent /usr/local/bin/clamav-alert.sh
```

Indítsuk újra a clamd-t:

```
$ sudo systemctl restart clamd@scan
```

> **NOTE:** A karanténba mozgatást már natívan támogatja az 1.2-es ClamAV, azonban a cikk írásakor ez még nem volt elérhető Fedora telepítő csomagként:
>
> ```
> $ clamd --version
> ClamAV 1.0.8/27590/Thu Mar 27 10:00:11 2025
> ```
>
> Az új verzióban az alábbi beállítások már elérhetőek: OnInfected, QuarantineDirectory

Tesztelés

Hozzunk létre egy virus fájlt a hivatalos teszt vírus tartalommal, valahol a /home/adam mappa alatt, amit figyel a clamonacc.

/home/adam/test-virus.txt

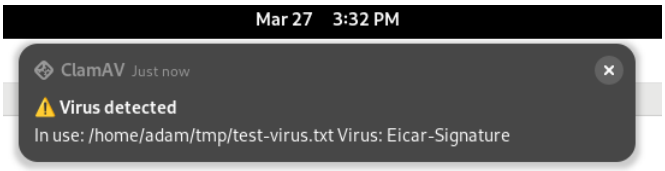
```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Ezt minden víruskereső ismeri.

Vírusos fájl: lockolt eset (nincs karantén)

Nyissuk meg egy grafikus szövegszerkesztővel a vírusos fájlt:

```
$ gedit /home/adam/tmp/test-virus.txt
```



Ekkor nem fogja tudni karanténba tenni, mert fogja a gedit a megnyitás alatt lévő fájlt.

> **IMPORTANT**: Amikor clamonacc egyszer már felismerte a vírust, nem engedi többé hozzáférni a fájlhoz, onnantól kezdve minden műveletet blokkolni fog vele, tehát ez a fájl már akkor sem fog tudni átkerülni karanténba, ha olyan műveletet végeznénk rajta, ami nem lockolja a fájlt, pl egy cat.

Vírusos fájl: lockolás nélküli művelet

Ilyen van?? TODO...

GUI használata

Tesztelés

Hogyan látom épp mit csinál a clamd

```
sudo strace -p $(pidof clamd | cut -d" " -f1) -s 200 -e trace=recvfrom,sendto
...
...
--- SIGPIPE {si_signo=SIGPIPE, si_code=SI_USER, si_pid=2038, si_uid=1000} ---
sendto(5, "<183>Mar 26 00:24:48 clamd[2038]: Client disconnected (FD 11)", 61, MSG_NOSIGNAL, NULL, 0) = 61
sendto(11, "", 0, 0, NULL, 0)           = -1 EPIPE (Broken pipe)
--- SIGPIPE {si_signo=SIGPIPE, si_code=SI_USER, si_pid=2038, si_uid=1000} ---
sendto(5, "<183>Mar 26 00:24:48 clamd[2038]: Client disconnected (FD 11)", 61, MSG_NOSIGNAL, NULL, 0) = 61
```

Honnan látom, hogy tényleg átvizsgálta e a megnyitott fájlt?

Példa vírus letöltése:

TODO: mi történik ha vírust talál,

hogy tudom megnézni, hogy mi történt, hogy működik e stb.

Troubleshooting

.....

ez mit jelent?

Mar 24 16:55:04 fedora clamd[1912]: SWF support enabled.
Mar 24 16:55:04 fedora clamd[1912]: HTML support enabled.
Mar 24 16:55:04 fedora clamd[1912]: XMLDOCS support enabled.
Mar 24 16:55:04 fedora clamd[1912]: HWP3 support enabled.
Mar 24 16:55:04 fedora clamd[1912]: Self checking every 600 seconds.