Guia de Instalação Greenbone Community Edition (Building From Source)

Autor: Narcelio Soares Lima

Mentor: Max Eduardo Vizcarra Melgar

Última revisão: Julho de 2025

Sumário

1. Configurações do Ambiente e Pré-requisitos

Este tutorial foi elaborado com base no processo oficial de construção a partir do código-fonte (Building from Source) mantido pela Greenbone. Seu objetivo é fornecer um roteiro modular e objetivo para a instalação da Greenbone Community Edition, com ênfase na clareza de cada etapa e no entendimento do fluxo de instalação do início ao fim.

Esta documentação foi projetada para servir como uma referência padronizada no contexto dos ambientes do PoP (Ponto de Presença da RNP), garantindo uniformidade na instalação da solução de análise de vulnerabilidades em dispositivos que atendam aos requisitos mínimos definidos. O conteúdo foi cuidadosamente estruturado para privilegiar a rastreabilidade dos comandos, a separação lógica por componentes, e a coerência com práticas recomendadas de compilação e provisionamento de serviços.

📚 Nível de dificuldade: intermediário a avançado

Recomenda-se que o executor tenha familiaridade com ambiente Linux, compilação de pacotes a partir do código-fonte, controle de permissões e administração de serviços via systemd.

1.1 Ambiente utilizado para testes

A instalação foi realizada em um ambiente controlado com as seguintes características mínimas:

Recurso	Configuração/Versão
CPU	4 núcleos físicos
Memória RAM	6 GB
SWAP	8 GB
Armazenamento livre	60 GB
Sistema operacional	Ubuntu 22.04 LTS
Acesso privilegiado	Usuário com `sudo` habilitado

⚠️ Versões diferentes do Ubuntu ou de outras distribuições Linux podem requerer ajustes nos pacotes, caminhos ou permissões. Este tutorial considera que o ambiente já possui as configurações básicas e os pacotes essenciais devidamente ajustados.

Atendidos os requisitos descritos, siga para a próxima etapa da instalação.

1.2 Versões dos componentes

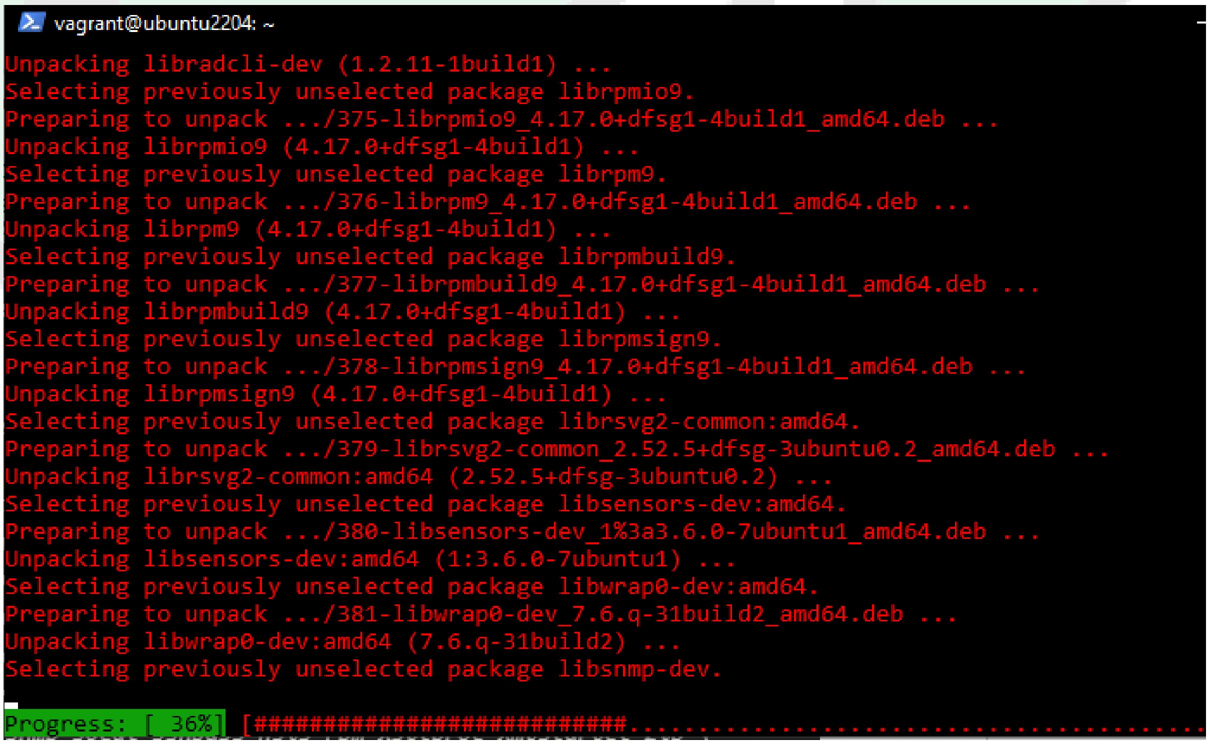
Este tutorial foi validado utilizando as seguintes versões dos principais componentes do Greenbone Community Edition:

Componente	Versão
GVM Libraries	22.22.0
Greenbone Manager (gvmd)	26.0.0
PostgreSQL Adapter (pg-gvm)	22.6.9
Greenbone Security Assistant (GSA)	25.0.0
GSAD (Web Daemon)	24.3.0
openvas-smb	22.5.3
openvas-scanner	23.21.0
ospd-openvas	22.9.0
openvasd (daemon em Rust)	25.0.0

1.3 Preparação do ambiente

Passo 1 - Instale todas as bibliotecas e pacotes que serão necessários para realizar o procedimento de instalação do Greenbone.

```
sudo apt update && sudo apt install --assume-yes \
  build-essential cmake pkg-config curl wget git gnupg \
  python3 python3-pip python3-venv \
  libglib2.0-dev libgnutls28-dev libgpgme-dev libgcrypt20-dev \
  libhiredis-dev libksba-dev libpcap-dev libssh-dev libnet1-dev libldap2-dev \
  libpaho-mqtt-dev libical-dev libxml2-dev libxml2-utils libjson-glib-dev
libcjson-dev \
  libssl-dev libcurl4-gnutls-dev libbrotli-dev libmicrohttpd-dev \
  libpq-dev postgresql postgresql-server-dev-all \
  libsnmp-dev libradcli-dev libpopt-dev libunistring-dev libbsd-dev \
  libnetfilter-queue-dev libsystemd-dev libjson-glib-dev uuid-dev \
  dpkg fakeroot bison \
  clang clang-format doxygen graphviz \
  nmap smbclient snmp socat sshpass nsis rpm xsltproc xmlstarlet zip \
  gnutls-bin gpgsm openssh-client rsync \
  gcc-mingw-w64 perl-base heimdal-multidev krb5-multidev \
  python3-setuptools python3-packaging python3-wrapt \
  python3-cffi python3-psutil python3-defusedxml \
  python3-paramiko python3-redis python3-gnupg \
  python3-paho-mqtt python3-impacket \
  texlive-fonts-recommended texlive-latex-extra \
  python3-lxml autoconf libtool \
  zlib1g-dev libpsl-dev\
```

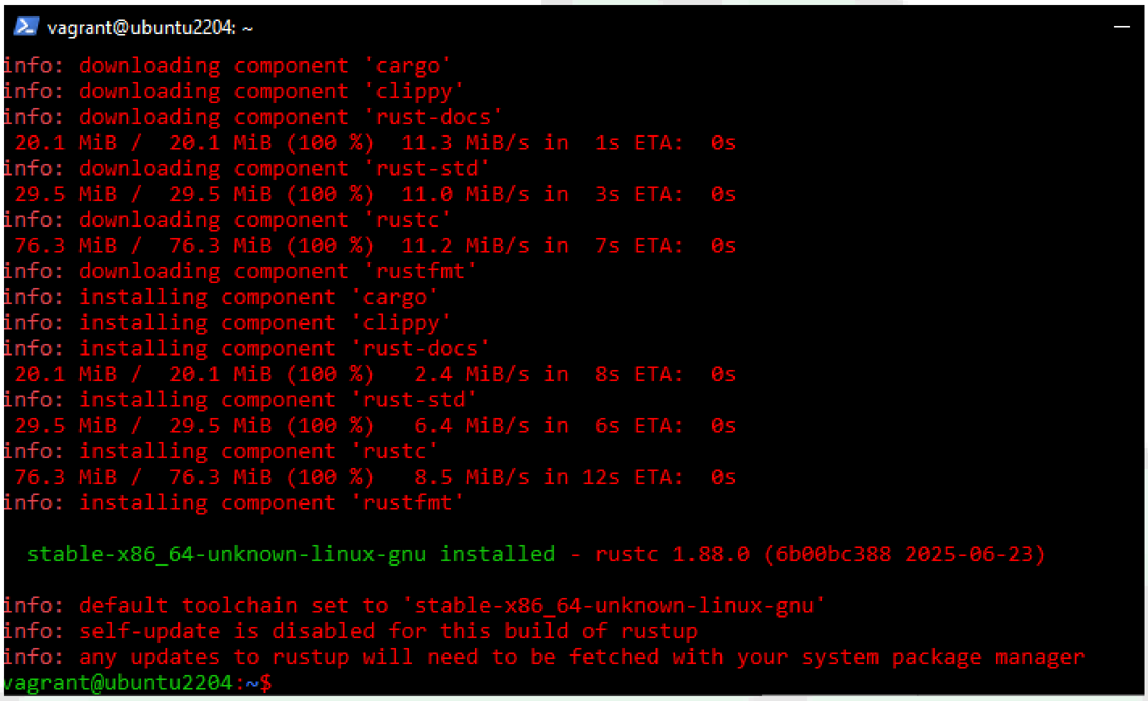


```
vagrant@ubuntu2204: ~
Unpacking libradcli-dev (1.2.11-1build1) ...
Selecting previously unselected package librpmio9.
Preparing to unpack .../375-librpmio9_4.17.0+dfsg1-4build1_amd64.deb ...
Unpacking librpmio9 (4.17.0+dfsg1-4build1) ...
Selecting previously unselected package librpm9.
Preparing to unpack .../376-librpm9_4.17.0+dfsg1-4build1_amd64.deb ...
Unpacking librpm9 (4.17.0+dfsg1-4build1) ...
Selecting previously unselected package librpmbuild9.
Preparing to unpack .../377-librpmbuild9_4.17.0+dfsg1-4build1_amd64.deb ...
Unpacking librpmbuild9 (4.17.0+dfsg1-4build1) ...
Selecting previously unselected package librpmsign9.
Preparing to unpack .../378-librpmsign9_4.17.0+dfsg1-4build1_amd64.deb ...
Unpacking librpmsign9 (4.17.0+dfsg1-4build1) ...
Selecting previously unselected package librsvg2-common:amd64.
Preparing to unpack .../379-librsvg2-common_2.52.5+dfsg-3ubuntu0.2_amd64.deb ...
Unpacking librsvg2-common:amd64 (2.52.5+dfsg-3ubuntu0.2) ...
Selecting previously unselected package libsensors-dev:amd64.
Preparing to unpack .../380-libsensors-dev_1%3a3.6.0-7ubuntu1_amd64.deb ...
Unpacking libsensors-dev:amd64 (1:3.6.0-7ubuntu1) ...
Selecting previously unselected package libwrap0-dev:amd64.
Preparing to unpack .../381-libwrap0-dev_7.6.q-31build2_amd64.deb ...
Unpacking libwrap0-dev:amd64 (7.6.q-31build2) ...
Selecting previously unselected package libsnmp-dev.

Progress: [ 36%] [#############################.......................................]
```

Figura 1: *Instalação de bibliotecas necessárias para o Greenbone.*

Passo 2 - Realize a instalação do Rust via Snap e atualize o ambiente para utilizar a versão mais recente dos componentes.

```
sudo snap install rustup --classic && rustup update stable
```

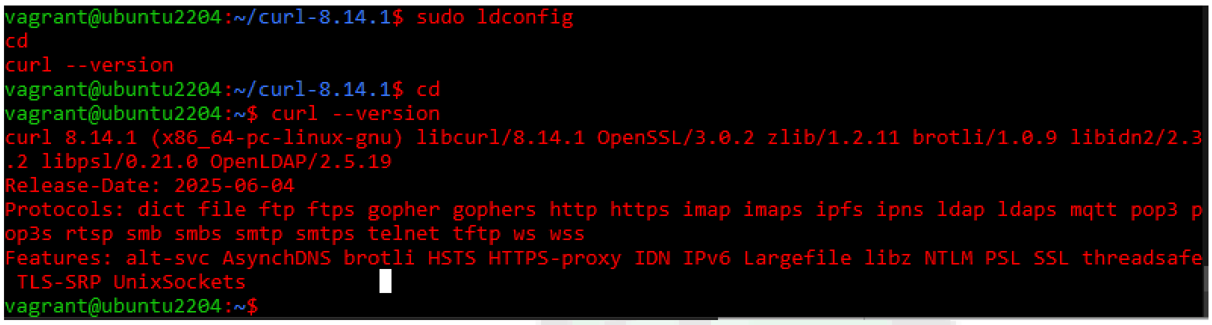


```
vagrant@ubuntu2204: ~
info: downloading component 'cargo'
info: downloading component 'clippy'
info: downloading component 'rust-docs'
 20.1 MiB /  20.1 MiB (100 %)  11.3 MiB/s in  1s ETA:  0s
info: downloading component 'rust-std'
 29.5 MiB /  29.5 MiB (100 %)  11.0 MiB/s in  3s ETA:  0s
info: downloading component 'rustc'
 76.3 MiB /  76.3 MiB (100 %)  11.2 MiB/s in  7s ETA:  0s
info: downloading component 'rustfmt'
info: installing component 'cargo'
info: installing component 'clippy'
info: installing component 'rust-docs'
 20.1 MiB /  20.1 MiB (100 %)   2.4 MiB/s in  8s ETA:  0s
info: installing component 'rust-std'
 29.5 MiB /  29.5 MiB (100 %)   6.4 MiB/s in  6s ETA:  0s
info: installing component 'rustc'
 76.3 MiB /  76.3 MiB (100 %)   8.5 MiB/s in 12s ETA:  0s
info: installing component 'rustfmt'

  stable-x86_64-unknown-linux-gnu installed - rustc 1.88.0 (6b00bc388 2025-06-23)

info: default toolchain set to 'stable-x86_64-unknown-linux-gnu'
info: self-update is disabled for this build of rustup
info: any updates to rustup will need to be fetched with your system package manager
vagrant@ubuntu2204:~$
```

Figura 2: Finalização da configuração do rust.

Passo 3 - Instale manualmente a versão mais recente do Curl e, ao finalizar o processo de compilação, utilize o comando curl --version para verificar se a instalação foi concluída com sucesso e a versão está correta.

```
wget https://curl.se/download/curl-8.14.1.tar.gz
tar -xvzf curl-8.14.1.tar.gz
cd curl-8.14.1

./configure --with-ssl

make -j$(nproc)
sudo make install

sudo ldconfig

cd
curl --version
```



```
vagrant@ubuntu2204:~/curl-8.14.1$ sudo ldconfig
cd
curl --version
vagrant@ubuntu2204:~/curl-8.14.1$ cd
vagrant@ubuntu2204:~$ curl --version
curl 8.14.1 (x86_64-pc-linux-gnu) libcurl/8.14.1 OpenSSL/3.0.2 zlib/1.2.11 brotli/1.0.9 libidn2/2.3
.2 libpsl/0.21.0 OpenLDAP/2.5.19
Release-Date: 2025-06-04
Protocols: dict file ftp ftps gopher gophers http https imap imaps ipfs ipns ldap ldaps mqtt pop3 p
op3s rtsp smb smbs smtp smtps telnet tftp ws wss
Features: alt-svc AsynchDNS brotli HSTS HTTPS-proxy IDN IPv6 Largefile libz NTLM PSL SSL threadsafe
 TLS-SRP UnixSockets
vagrant@ubuntu2204:~$
```

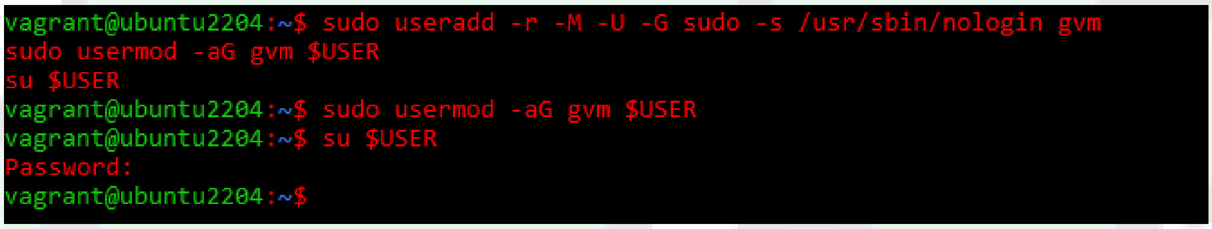
Figura 3: Verificação da versão atual do Curl no sistema.

Passo 4 - Crie o usuário gvm, atribua os privilégios “sudo”.

```
sudo useradd -r -M -U -G sudo -s /usr/sbin/nologin gvm && sudo usermod -aG gvm
$USER
```

Passo 5 - As mudanças entrarão em vigor ao fechar o terminal ou re-logar no usuário através do comando.

```
su $USER
```



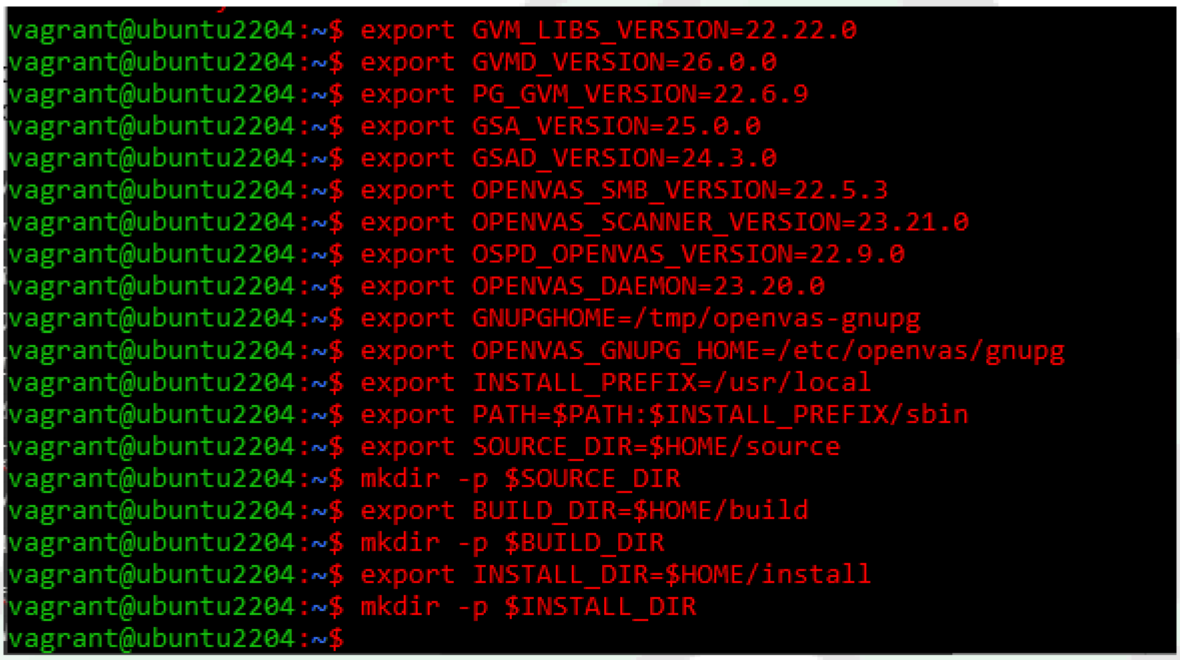
```
vagrant@ubuntu2204:~$ sudo useradd -r -M -U -G sudo -s /usr/sbin/nologin gvm
sudo usermod -aG gvm $USER
su $USER
vagrant@ubuntu2204:~$ sudo usermod -aG gvm $USER
vagrant@ubuntu2204:~$ su $USER
Password:
vagrant@ubuntu2204:~$
```

Figura 4: Criando e configurando usuário gvm e re-logando no usuário atual.

Passo 5 - Configure as variáveis do ambiente.

⚠️ Caso o terminal seja fechado ou a conexão SSH interrompida, as variáveis de ambiente serão perdidas. Será necessário reconfigurá-las para continuar a instalação.

```
export GVM_LIBS_VERSION=22.22.0
export GVMD_VERSION=26.0.0
export PG_GVM_VERSION=22.6.9
export GSA_VERSION=25.0.0
export GSAD_VERSION=24.3.0
export OPENVAS_SMB_VERSION=22.5.3
export OPENVAS_SCANNER_VERSION=23.21.0
export OSPD_OPENVAS_VERSION=22.9.0
export OPENVAS_DAEMON=23.21.0
export GNUPGHOME=/tmp/openvas-gnupg
export OPENVAS_GNUPG_HOME=/etc/openvas/gnupg
export INSTALL_PREFIX=/usr/local
export PATH=$PATH:$INSTALL_PREFIX/sbin
export SOURCE_DIR=$HOME/source
mkdir -p $SOURCE_DIR
export BUILD_DIR=$HOME/build
mkdir -p $BUILD_DIR
export INSTALL_DIR=$HOME/install
mkdir -p $INSTALL_DIR
```



```
vagrant@ubuntu2204:~$ export GVM_LIBS_VERSION=22.22.0
vagrant@ubuntu2204:~$ export GVMD_VERSION=26.0.0
vagrant@ubuntu2204:~$ export PG_GVM_VERSION=22.6.9
vagrant@ubuntu2204:~$ export GSA_VERSION=25.0.0
vagrant@ubuntu2204:~$ export GSAD_VERSION=24.3.0
vagrant@ubuntu2204:~$ export OPENVAS_SMB_VERSION=22.5.3
vagrant@ubuntu2204:~$ export OPENVAS_SCANNER_VERSION=23.21.0
vagrant@ubuntu2204:~$ export OSPD_OPENVAS_VERSION=22.9.0
vagrant@ubuntu2204:~$ export OPENVAS_DAEMON=23.20.0
vagrant@ubuntu2204:~$ export GNUPGHOME=/tmp/openvas-gnupg
vagrant@ubuntu2204:~$ export OPENVAS_GNUPG_HOME=/etc/openvas/gnupg
vagrant@ubuntu2204:~$ export INSTALL_PREFIX=/usr/local
vagrant@ubuntu2204:~$ export PATH=$PATH:$INSTALL_PREFIX/sbin
vagrant@ubuntu2204:~$ export SOURCE_DIR=$HOME/source
vagrant@ubuntu2204:~$ mkdir -p $SOURCE_DIR
vagrant@ubuntu2204:~$ export BUILD_DIR=$HOME/build
vagrant@ubuntu2204:~$ mkdir -p $BUILD_DIR
vagrant@ubuntu2204:~$ export INSTALL_DIR=$HOME/install
vagrant@ubuntu2204:~$ mkdir -p $INSTALL_DIR
vagrant@ubuntu2204:~$
```

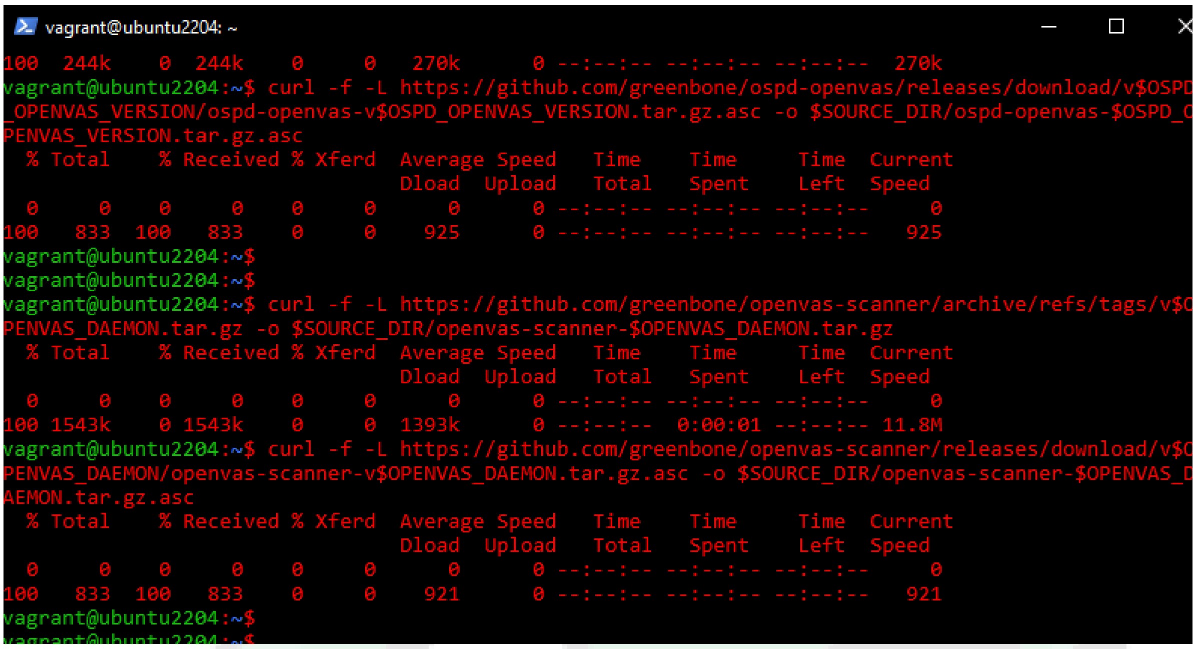
Figura 5: Configurando variáveis de ambiente.

Passo 6 - Pronto, com o ambiente devidamente preparado, siga para o próximo passo da instalação dos componentes.

2. Baixando e verificando a integridade dos arquivos

Passo 1 – Copie e cole todo o código abaixo no terminal para fazer o download do tarball e das respectivas chaves de assinatura dos pacotes do Greenbone Community Edition.

```
for module in \
  "gvm-libs:$GVM_LIBS_VERSION" \
  "gvmd:$GVMD_VERSION" \
  "pg-gvm:$PG_GVM_VERSION" \
  "gsa:$GSA_VERSION:gsa-dist" \
  "gsad:$GSAD_VERSION" \
  "openvas-smb:$OPENVAS_SMB_VERSION:v" \
  "openvas-scanner:$OPENVAS_SCANNER_VERSION:v" \
  "ospd-openvas:$OSPD_OPENVAS_VERSION:v" \
  "openvas-scanner:$OPENVAS_DAEMON:v"; do

  IFS=":" read -r name ver asc_type <<< "$module"

  if [[ "$asc_type" == "gsa-dist" ]]; then
    archive="gsa-dist-$ver"
    output="gsa-$ver"

src_url="https://github.com/greenbone/$name/releases/download/v$ver/$archive.t
ar.gz"

asc_url="https://github.com/greenbone/$name/releases/download/v$ver/$archive.t
ar.gz.asc"
  elif [[ "$asc_type" == "v" ]]; then
    archive="$name-$ver"
    output="$archive"

src_url="https://github.com/greenbone/$name/archive/refs/tags/v$ver.tar.gz"

asc_url="https://github.com/greenbone/$name/releases/download/v$ver/$name-v$ve
r.tar.gz.asc"
  else
    archive="$name-$ver"
    output="$archive"

src_url="https://github.com/greenbone/$name/archive/refs/tags/v$ver.tar.gz"

asc_url="https://github.com/greenbone/$name/releases/download/v$ver/$name-$ver
.tar.gz.asc"
  fi

  echo "Fonte: $src_url"
  curl -fL "$src_url" -o "$SOURCE_DIR/$output.tar.gz"

  echo "Assinatura: $asc_url"
  curl -fL "$asc_url" -o "$SOURCE_DIR/$output.tar.gz.asc"

  echo "----------------------------------------------"
done
```

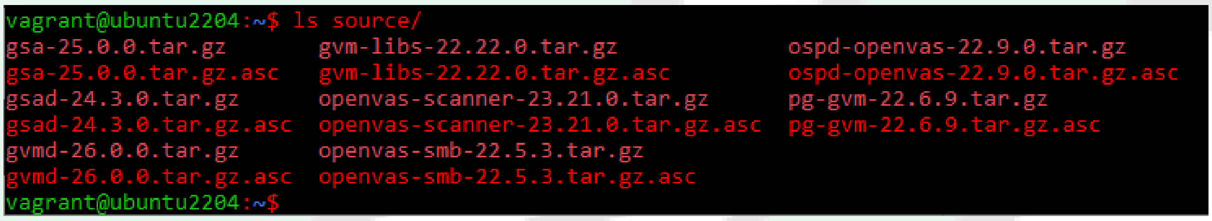


```
100  244k    0  244k    0     0   270k      0 --:--:-- --:--:-- --:--:--  270k
vagrant@ubuntu2204:~$ curl -f -L https://github.com/greenbone/ospd-openvas/releases/download/v$OSPD
_OPENVAS_VERSION/ospd-openvas-v$OSPD_OPENVAS_VERSION.tar.gz.asc -o $SOURCE_DIR/ospd-openvas-$OSPD_O
PENVAS_VERSION.tar.gz.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   833  100   833    0     0    925      0 --:--:-- --:--:-- --:--:--   925
vagrant@ubuntu2204:~$
vagrant@ubuntu2204:~$
vagrant@ubuntu2204:~$ curl -f -L https://github.com/greenbone/openvas-scanner/archive/refs/tags/v$O
PENVAS_DAEMON.tar.gz -o $SOURCE_DIR/openvas-scanner-$OPENVAS_DAEMON.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 1543k    0 1543k    0     0  1393k      0 --:--:--  0:00:01 --:--:-- 11.8M
vagrant@ubuntu2204:~$ curl -f -L https://github.com/greenbone/openvas-scanner/releases/download/v$O
PENVAS_DAEMON/openvas-scanner-v$OPENVAS_DAEMON.tar.gz.asc -o $SOURCE_DIR/openvas-scanner-$OPENVAS_D
AEMON.tar.gz.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100   833  100   833    0     0    921      0 --:--:-- --:--:-- --:--:--   921
vagrant@ubuntu2204:~$
```

Figura 6: Download dos arquivos contendo o código fonte do Greenbone.

Passo 2 - Verifique se os tarballs foram baixados corretamente conferindo o conteúdo da pasta source.

```
ls source/
```

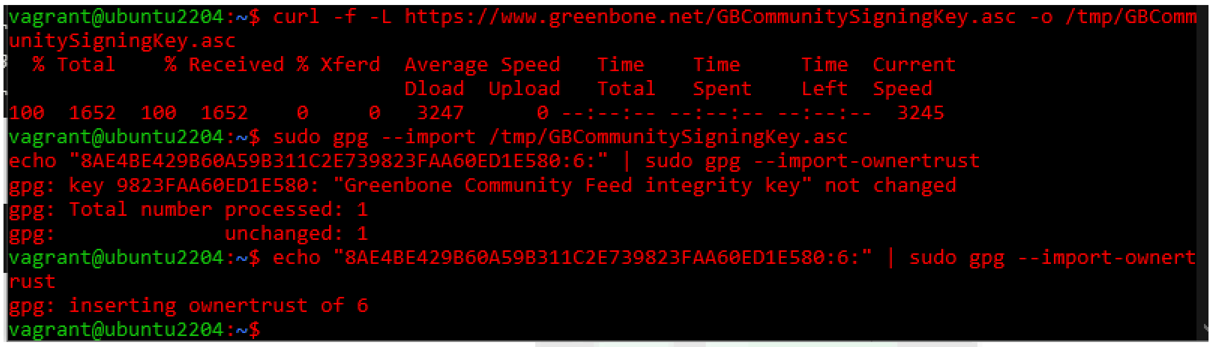


```
vagrant@ubuntu2204:~$ ls source/
gsa-25.0.0.tar.gz        gvm-libs-22.22.0.tar.gz             ospd-openvas-22.9.0.tar.gz
gsa-25.0.0.tar.gz.asc    gvm-libs-22.22.0.tar.gz.asc         ospd-openvas-22.9.0.tar.gz.asc
gsad-24.3.0.tar.gz       openvas-scanner-23.21.0.tar.gz      pg-gvm-22.6.9.tar.gz
gsad-24.3.0.tar.gz.asc   openvas-scanner-23.21.0.tar.gz.asc  pg-gvm-22.6.9.tar.gz.asc
gvmd-26.0.0.tar.gz       openvas-smb-22.5.3.tar.gz
gvmd-26.0.0.tar.gz.asc   openvas-smb-22.5.3.tar.gz.asc
vagrant@ubuntu2204:~$
```

Figura 7: Verificando arquivos baixados através do comando "ls".

Passo 3 - Para conferir se os arquivos estão funcionais, importe a chave de verificação do Greenbone.

```
curl -f -L https://www.greenbone.net/GBCommunitySigningKey.asc -o
/tmp/GBCommunitySigningKey.asc

sudo gpg --import /tmp/GBCommunitySigningKey.asc

echo "8AE4BE429B60A59B311C2E739823FAA60ED1E580:6:" | sudo gpg
--import-ownertrust
```

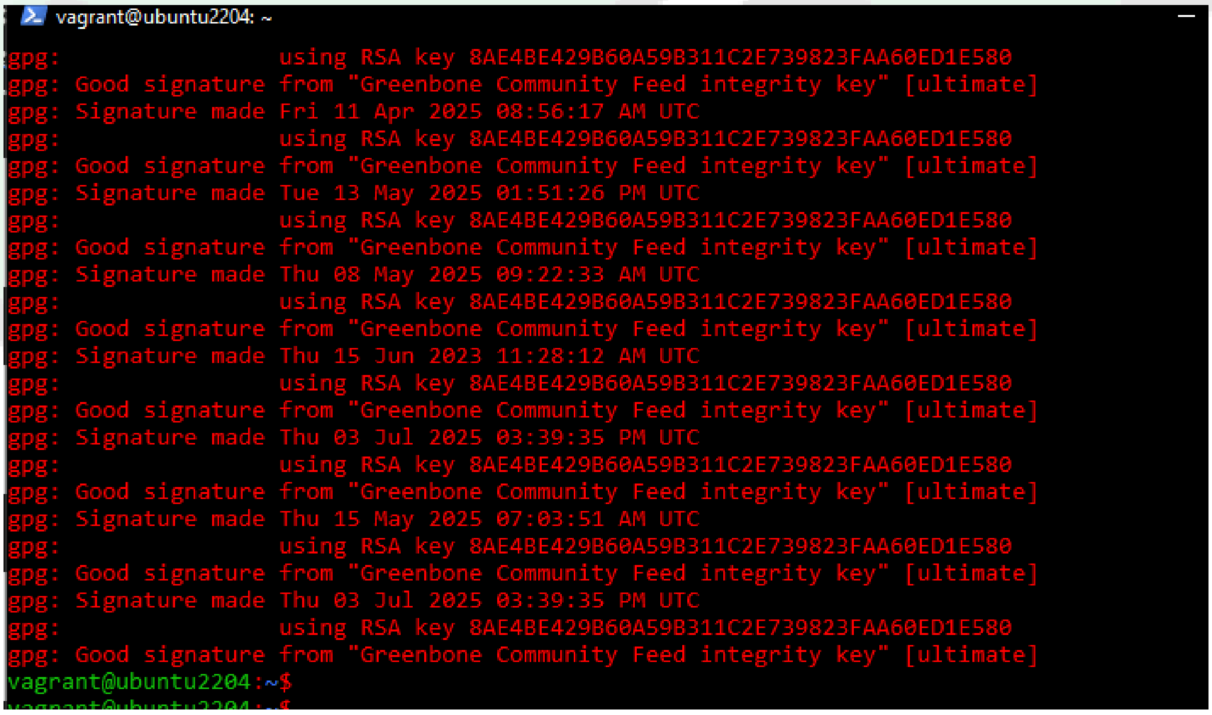


```
vagrant@ubuntu2204:~$ curl -f -L https://www.greenbone.net/GBCommunitySigningKey.asc -o /tmp/GBComm
unitySigningKey.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1652  100  1652    0     0   3247      0 --:--:-- --:--:-- --:--:--  3245
vagrant@ubuntu2204:~$ sudo gpg --import /tmp/GBCommunitySigningKey.asc
echo "8AE4BE429B60A59B311C2E739823FAA60ED1E580:6:" | sudo gpg --import-ownertrust
gpg: key 9823FAA60ED1E580: "Greenbone Community Feed integrity key" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
vagrant@ubuntu2204:~$ echo "8AE4BE429B60A59B311C2E739823FAA60ED1E580:6:" | sudo gpg --import-ownert
rust
gpg: inserting ownertrust of 6
vagrant@ubuntu2204:~$
```

Figura 8: Baixando e configurando a chave de integridade do Greenbone.

Passo 4 - Em seguida, verifique a validade das assinaturas dos arquivos baixados.

```
for pkg in \
  gvm-libs-$GVM_LIBS_VERSION \
  gvmd-$GVMD_VERSION \
  pg-gvm-$PG_GVM_VERSION \
  gsa-$GSA_VERSION \
  gsad-$GSAD_VERSION \
  openvas-smb-$OPENVAS_SMB_VERSION \
  openvas-scanner-$OPENVAS_SCANNER_VERSION \
  ospd-openvas-$OSPD_OPENVAS_VERSION \
  openvas-scanner-$OPENVAS_DAEMON; do
  sudo gpg --verify "$SOURCE_DIR/$pkg.tar.gz.asc" "$SOURCE_DIR/$pkg.tar.gz"
done
```



```
vagrant@ubuntu2204: ~
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Fri 11 Apr 2025 08:56:17 AM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Tue 13 May 2025 01:51:26 PM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Thu 08 May 2025 09:22:33 AM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Thu 15 Jun 2023 11:28:12 AM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Thu 03 Jul 2025 03:39:35 PM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Thu 15 May 2025 07:03:51 AM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
gpg: Signature made Thu 03 Jul 2025 03:39:35 PM UTC
gpg:                using RSA key 8AE4BE429B60A59B311C2E739823FAA60ED1E580
gpg: Good signature from "Greenbone Community Feed integrity key" [ultimate]
vagrant@ubuntu2204:~$
vagrant@ubuntu2204:~$
```

Figura 9: Verificando a integridade dos arquivos de código fonte do Greenbone.

3. Compilando e Instalando os módulos

3.1 Greenbone Vulnerability Manager

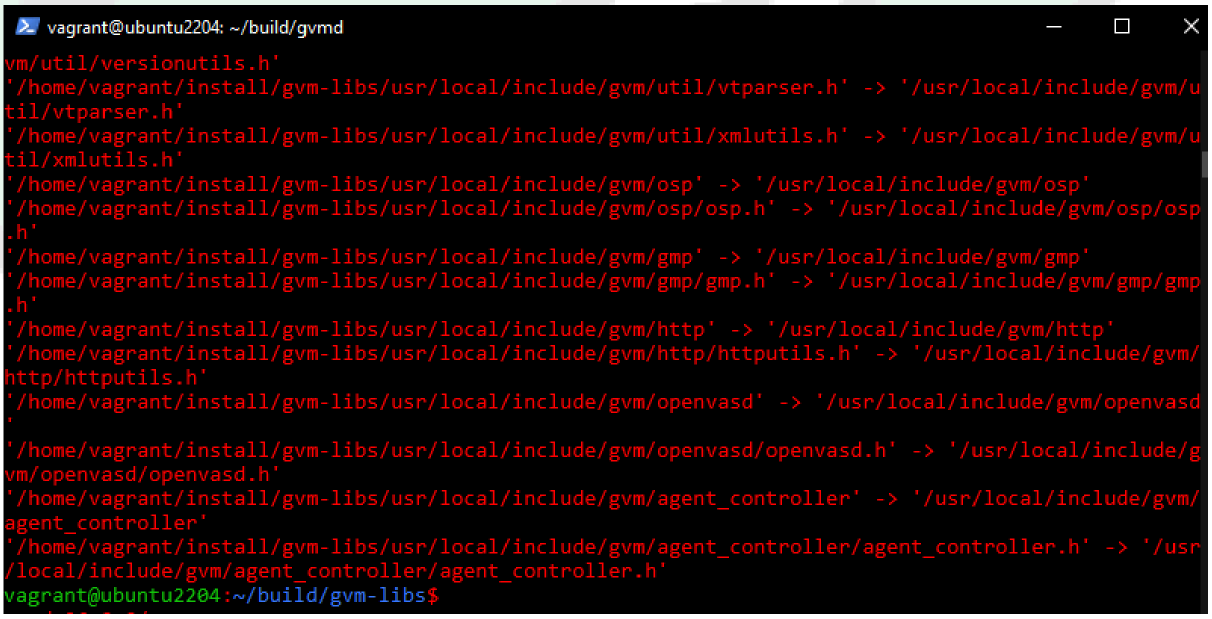
O Greenbone Vulnerability Manager (GVM) é um framework de código aberto utilizado para identificar, gerenciar e reportar vulnerabilidades em sistemas e redes. Ele é composto por módulos integrados como gvmd (gerenciador central), gvm-libs (bibliotecas compartilhadas) e postgres-gvm (extensões específicas para banco de dados).

3.1.1 gvm-libs

O gvm-libs é uma biblioteca-base compartilhada entre os módulos do GVM, fornece funcionalidades comuns para comunicação, gerenciamento e tratamento de dados.

Passo 1 - Agora, com todos os arquivos baixados da etapa anterior, prossiga com a compilação e a instalação do gvm-libs.

```
tar -C "$SOURCE_DIR" -xvzf "$SOURCE_DIR/gvm-libs-$GVM_LIBS_VERSION.tar.gz" &&
mkdir -p "$BUILD_DIR/gvm-libs" && cmake -S
"$SOURCE_DIR/gvm-libs-$GVM_LIBS_VERSION" -B "$BUILD_DIR/gvm-libs"
-DCMAKE_INSTALL_PREFIX="$INSTALL_PREFIX" -DCMAKE_BUILD_TYPE=Release
-DSYSCONFDIR=/etc -DLOCALSTATEDIR=/var && cmake --build "$BUILD_DIR/gvm-libs"
-j"$(nproc)" && mkdir -p "$INSTALL_DIR/gvm-libs" && cd "$BUILD_DIR/gvm-libs"
&& make DESTDIR="$INSTALL_DIR/gvm-libs" install && sudo cp -rv
"$INSTALL_DIR/gvm-libs"/* /
```



```
vm/util/versionutils.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/util/vtparser.h' -> '/usr/local/include/gvm/u
til/vtparser.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/util/xmlutils.h' -> '/usr/local/include/gvm/u
til/xmlutils.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/osp' -> '/usr/local/include/gvm/osp'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/osp/osp.h' -> '/usr/local/include/gvm/osp/osp
.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/gmp' -> '/usr/local/include/gvm/gmp'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/gmp/gmp.h' -> '/usr/local/include/gvm/gmp/gmp
.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/http' -> '/usr/local/include/gvm/http'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/http/httputils.h' -> '/usr/local/include/gvm/
http/httputils.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/openvasd' -> '/usr/local/include/gvm/openvasd
'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/openvasd/openvasd.h' -> '/usr/local/include/g
vm/openvasd/openvasd.h'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/agent_controller' -> '/usr/local/include/gvm/
agent_controller'
'/home/vagrant/install/gvm-libs/usr/local/include/gvm/agent_controller/agent_controller.h' -> '/usr
/local/include/gvm/agent_controller/agent_controller.h'
vagrant@ubuntu2204:~/build/gvm-libs$
```

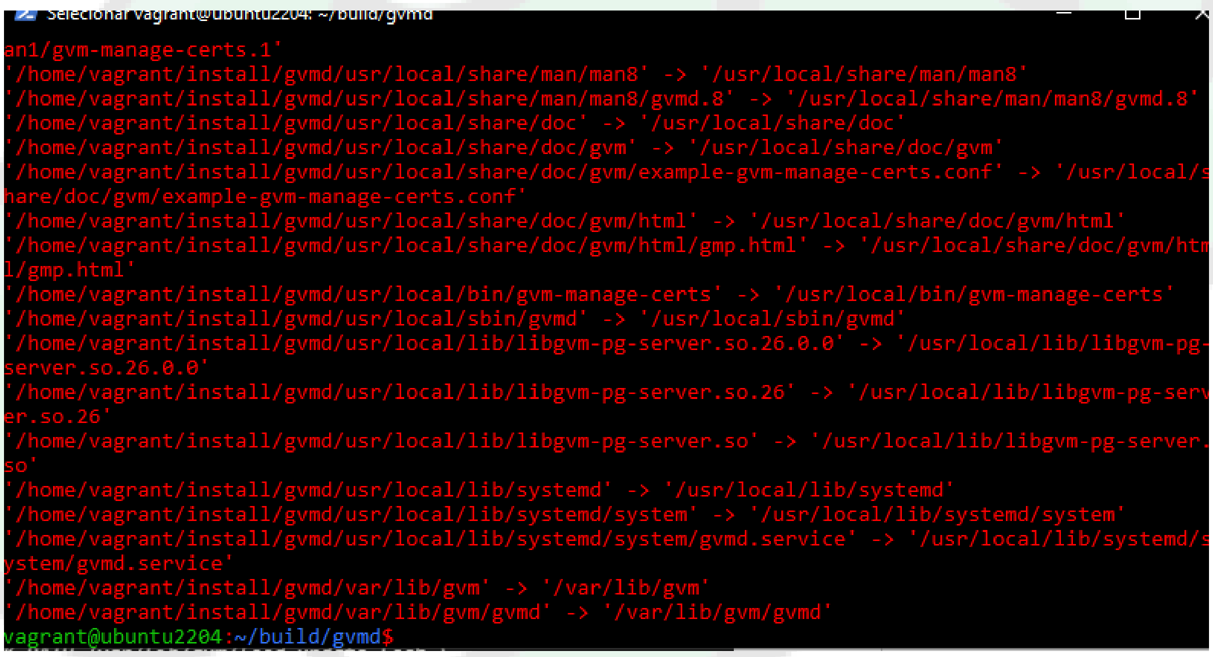
Figura 10: Instalação e transferência do gvm-lib para pasta padrão do sistema.

3.1.2 gvmd

Gerenciador central de vulnerabilidades; coordena os scans, administra relatórios e interage com o banco de dados.

Passo 2 - Continue com a compilação dos demais módulos, agora o gvmd, copie o cole o código abaixo no terminal.

```
tar -C "$SOURCE_DIR" -xvzf "$SOURCE_DIR/gvmd-$GVMD_VERSION.tar.gz" && mkdir -p
"$BUILD_DIR/gvmd" && cmake -S "$SOURCE_DIR/gvmd-$GVMD_VERSION" -B
"$BUILD_DIR/gvmd" -DCMAKE_INSTALL_PREFIX="$INSTALL_PREFIX"
-DCMAKE_BUILD_TYPE=Release -DLOCALSTATEDIR=/var -DSYSCONFDIR=/etc
-DGVM_DATA_DIR=/var -DGVM_LOG_DIR=/var/log/gvm -DGVMD_RUN_DIR=/run/gvmd
-DOPENVAS_DEFAULT_SOCKET=/run/ospd/ospd-openvas.sock
-DGVM_FEED_LOCK_PATH=/var/lib/gvm/feed-update.lock
-DLOGROTATE_DIR=/etc/logrotate.d && cmake --build "$BUILD_DIR/gvmd"
-j"$(nproc)" && mkdir -p "$INSTALL_DIR/gvmd" && cd "$BUILD_DIR/gvmd" && make
DESTDIR="$INSTALL_DIR/gvmd" install && sudo cp -rv "$INSTALL_DIR/gvmd"/* /
```



```
Selecionar vagrant@ubuntu2204: ~/build/gvmd
an1/gvm-manage-certs.1'
'/home/vagrant/install/gvmd/usr/local/share/man/man8' -> '/usr/local/share/man/man8'
'/home/vagrant/install/gvmd/usr/local/share/man/man8/gvmd.8' -> '/usr/local/share/man/man8/gvmd.8'
'/home/vagrant/install/gvmd/usr/local/share/doc' -> '/usr/local/share/doc'
'/home/vagrant/install/gvmd/usr/local/share/doc/gvm' -> '/usr/local/share/doc/gvm'
'/home/vagrant/install/gvmd/usr/local/share/doc/gvm/example-gvm-manage-certs.conf' -> '/usr/local/s
hare/doc/gvm/example-gvm-manage-certs.conf'
'/home/vagrant/install/gvmd/usr/local/share/doc/gvm/html' -> '/usr/local/share/doc/gvm/html'
'/home/vagrant/install/gvmd/usr/local/share/doc/gvm/html/gmp.html' -> '/usr/local/share/doc/gvm/htm
l/gmp.html'
'/home/vagrant/install/gvmd/usr/local/bin/gvm-manage-certs' -> '/usr/local/bin/gvm-manage-certs'
'/home/vagrant/install/gvmd/usr/local/sbin/gvmd' -> '/usr/local/sbin/gvmd'
'/home/vagrant/install/gvmd/usr/local/lib/libgvm-pg-server.so.26.0.0' -> '/usr/local/lib/libgvm-pg-
server.so.26.0.0'
'/home/vagrant/install/gvmd/usr/local/lib/libgvm-pg-server.so.26' -> '/usr/local/lib/libgvm-pg-serv
er.so.26'
'/home/vagrant/install/gvmd/usr/local/lib/libgvm-pg-server.so' -> '/usr/local/lib/libgvm-pg-server.
so'
'/home/vagrant/install/gvmd/usr/local/lib/systemd' -> '/usr/local/lib/systemd'
'/home/vagrant/install/gvmd/usr/local/lib/systemd/system' -> '/usr/local/lib/systemd/system'
'/home/vagrant/install/gvmd/usr/local/lib/systemd/system/gvmd.service' -> '/usr/local/lib/systemd/s
ystem/gvmd.service'
'/home/vagrant/install/gvmd/var/lib/gvm' -> '/var/lib/gvm'
'/home/vagrant/install/gvmd/var/lib/gvm/gvmd' -> '/var/lib/gvm/gvmd'
vagrant@ubuntu2204:~/build/gvmd$
```

Figura 11: Instalação e transferência do gvmd para pasta padrão do sistema.

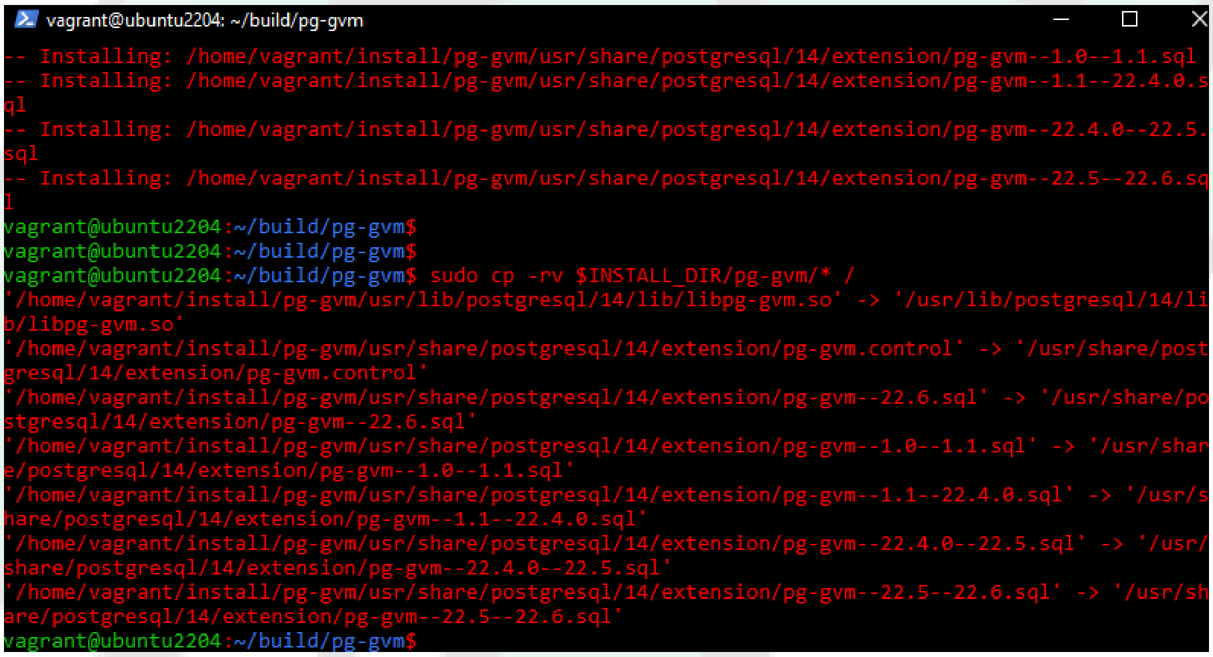
3.1.3 postgres-gvm

Extensão do PostgreSQL para o GVM; adiciona suporte específico à estrutura de dados usada pelo gvmd.

Passo 3 - Copie e cole o código a seguir no terminal, e aguarde o término.

‼ Fique atento às mensagens exibidas durante o processo de instalação. Em caso de erro, verifique se houve algum equívoco ao colar ou digitar os comandos no terminal.

```
tar -C "$SOURCE_DIR" -xvzf "$SOURCE_DIR/pg-gvm-$PG_GVM_VERSION.tar.gz" &&
mkdir -p "$BUILD_DIR/pg-gvm" && cmake -S "$SOURCE_DIR/pg-gvm-$PG_GVM_VERSION"
-B "$BUILD_DIR/pg-gvm" -DCMAKE_BUILD_TYPE=Release && cmake --build
"$BUILD_DIR/pg-gvm" -j"$(nproc)" && mkdir -p "$INSTALL_DIR/pg-gvm" && cd
"$BUILD_DIR/pg-gvm" && make DESTDIR="$INSTALL_DIR/pg-gvm" install && sudo cp
-rv "$INSTALL_DIR/pg-gvm"/* /
```



```
vagrant@ubuntu2204: ~/build/pg-gvm
-- Installing: /home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--1.0--1.1.sql
-- Installing: /home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--1.1--22.4.0.s
ql
-- Installing: /home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--22.4.0--22.5.
sql
-- Installing: /home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--22.5--22.6.sq
l
vagrant@ubuntu2204:~/build/pg-gvm$
vagrant@ubuntu2204:~/build/pg-gvm$
vagrant@ubuntu2204:~/build/pg-gvm$ sudo cp -rv $INSTALL_DIR/pg-gvm/* /
'/home/vagrant/install/pg-gvm/usr/lib/postgresql/14/lib/libpg-gvm.so' -> '/usr/lib/postgresql/14/li
b/libpg-gvm.so'
'/home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm.control' -> '/usr/share/post
gresql/14/extension/pg-gvm.control'
'/home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--22.6.sql' -> '/usr/share/po
stgresql/14/extension/pg-gvm--22.6.sql'
'/home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--1.0--1.1.sql' -> '/usr/shar
e/postgresql/14/extension/pg-gvm--1.0--1.1.sql'
'/home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--1.1--22.4.0.sql' -> '/usr/s
hare/postgresql/14/extension/pg-gvm--1.1--22.4.0.sql'
'/home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--22.4.0--22.5.sql' -> '/usr/
share/postgresql/14/extension/pg-gvm--22.4.0--22.5.sql'
'/home/vagrant/install/pg-gvm/usr/share/postgresql/14/extension/pg-gvm--22.5--22.6.sql' -> '/usr/sh
are/postgresql/14/extension/pg-gvm--22.5--22.6.sql'
vagrant@ubuntu2204:~/build/pg-gvm$
```

Figura 12: Instalação e transferência do postgres-gvm para pasta padrão do sistema.

3.2 Greenbone Security Assistant

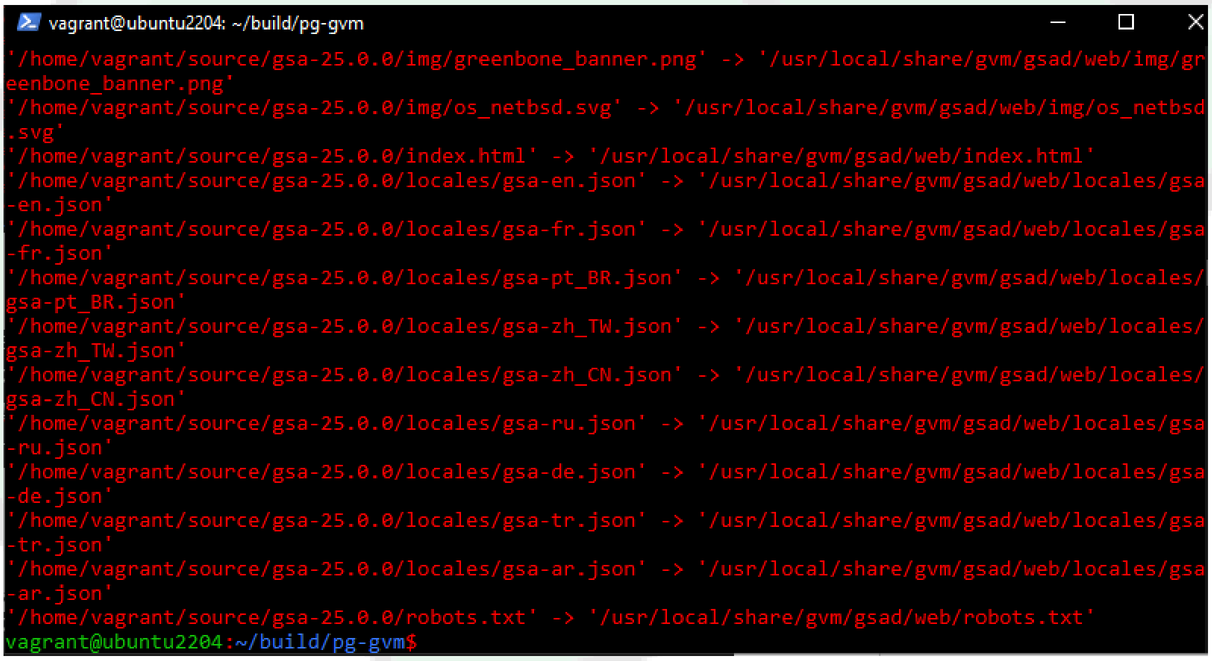
O Greenbone Security Assistant (GSA) é o front-end web do GVM, projetado para facilitar o gerenciamento de vulnerabilidades por meio de uma interface acessível via navegador. Ele é composto pelos módulos gsa (interface gráfica) e gsad (servidor de entrega HTTP). Juntos, permitem visualizar relatórios, configurar scans, e interagir com o gvmd sem necessidade de terminal.

3.2.1 gsa

Interface web do GVM baseada em JavaScript; permite ao usuário visualizar relatórios e operar os módulos por navegador.

Passo 4 - Copie e cole o código a seguir no terminal.

```
mkdir -p "$SOURCE_DIR/gsa-$GSA_VERSION" && tar -C
"$SOURCE_DIR/gsa-$GSA_VERSION" -xvzf "$SOURCE_DIR/gsa-$GSA_VERSION.tar.gz" &&
sudo mkdir -p "$INSTALL_PREFIX/share/gvm/gsad/web/" && sudo cp -rv
"$SOURCE_DIR/gsa-$GSA_VERSION"/* "$INSTALL_PREFIX/share/gvm/gsad/web/"
```



```
vagrant@ubuntu2204: ~/build/pg-gvm
'/home/vagrant/source/gsa-25.0.0/img/greenbone_banner.png' -> '/usr/local/share/gvm/gsad/web/img/greenbone_banner.png'
'/home/vagrant/source/gsa-25.0.0/img/os_netbsd.svg' -> '/usr/local/share/gvm/gsad/web/img/os_netbsd.svg'
'/home/vagrant/source/gsa-25.0.0/index.html' -> '/usr/local/share/gvm/gsad/web/index.html'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-en.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-en.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-fr.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-fr.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-pt_BR.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-pt_BR.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-zh_TW.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-zh_TW.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-zh_CN.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-zh_CN.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-ru.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-ru.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-de.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-de.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-tr.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-tr.json'
'/home/vagrant/source/gsa-25.0.0/locales/gsa-ar.json' -> '/usr/local/share/gvm/gsad/web/locales/gsa-ar.json'
'/home/vagrant/source/gsa-25.0.0/robots.txt' -> '/usr/local/share/gvm/gsad/web/robots.txt'
vagrant@ubuntu2204:~/build/pg-gvm$
```

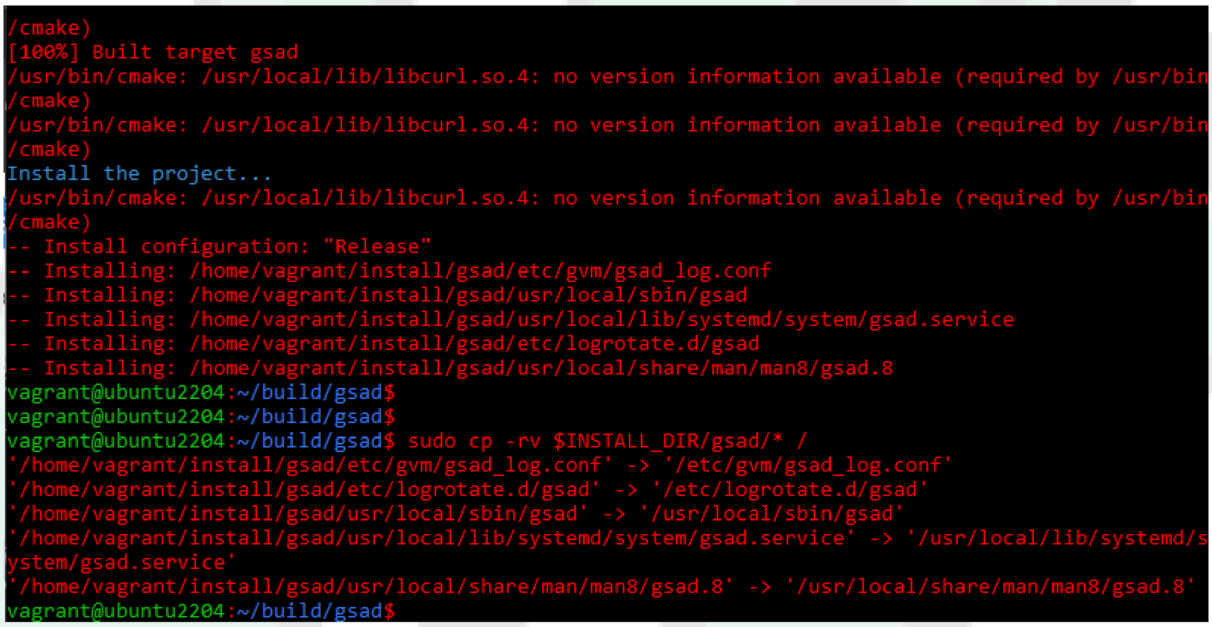
Figura 13: Instalação e transferência do gsa para pasta padrão do sistema.

3.2.2 gsad

Servidor HTTP responsável por entregar a interface GSA ao navegador; atua como ponte entre o front-end e o gerenciador gvmd.

Passo 5 - Copie e cole o código abaixo no terminal e aguarde a conclusão da instalação.

```
tar -C "$SOURCE_DIR" -xvzf "$SOURCE_DIR/gsad-$GSAD_VERSION.tar.gz" && mkdir -p
"$BUILD_DIR/gsad" && cmake -S "$SOURCE_DIR/gsad-$GSAD_VERSION" -B
"$BUILD_DIR/gsad" -DCMAKE_INSTALL_PREFIX="$INSTALL_PREFIX"
-DCMAKE_BUILD_TYPE=Release -DSYSCONFDIR=/etc -DLOCALSTATEDIR=/var
-DGVMD_RUN_DIR=/run/gvmd -DGSAD_RUN_DIR=/run/gsad -DGVM_LOG_DIR=/var/log/gvm
-DLOGROTATE_DIR=/etc/logrotate.d && cmake --build "$BUILD_DIR/gsad"
-j"$(nproc)" && mkdir -p "$INSTALL_DIR/gsad" && cd "$BUILD_DIR/gsad" && make
DESTDIR="$INSTALL_DIR/gsad" install && sudo cp -rv "$INSTALL_DIR/gsad"/* /
```



```
/cmake)
[100%] Built target gsad
/usr/bin/cmake: /usr/local/lib/libcurl.so.4: no version information available (required by /usr/bin
/cmake)
/usr/bin/cmake: /usr/local/lib/libcurl.so.4: no version information available (required by /usr/bin
/cmake)
Install the project...
/usr/bin/cmake: /usr/local/lib/libcurl.so.4: no version information available (required by /usr/bin
/cmake)
-- Install configuration: "Release"
-- Installing: /home/vagrant/install/gsad/etc/gvm/gsad_log.conf
-- Installing: /home/vagrant/install/gsad/usr/local/sbin/gsad
-- Installing: /home/vagrant/install/gsad/usr/local/lib/systemd/system/gsad.service
-- Installing: /home/vagrant/install/gsad/etc/logrotate.d/gsad
-- Installing: /home/vagrant/install/gsad/usr/local/share/man/man8/gsad.8
vagrant@ubuntu2204:~/build/gsad$
vagrant@ubuntu2204:~/build/gsad$
vagrant@ubuntu2204:~/build/gsad$ sudo cp -rv $INSTALL_DIR/gsad/* /
'/home/vagrant/install/gsad/etc/gvm/gsad_log.conf' -> '/etc/gvm/gsad_log.conf'
'/home/vagrant/install/gsad/etc/logrotate.d/gsad' -> '/etc/logrotate.d/gsad'
'/home/vagrant/install/gsad/usr/local/sbin/gsad' -> '/usr/local/sbin/gsad'
'/home/vagrant/install/gsad/usr/local/lib/systemd/system/gsad.service' -> '/usr/local/lib/systemd/s
ystem/gsad.service'
'/home/vagrant/install/gsad/usr/local/share/man/man8/gsad.8' -> '/usr/local/share/man/man8/gsad.8'
vagrant@ubuntu2204:~/build/gsad$
```

Figura 14: Instalação e transferência do gsad para pasta padrão do sistema.

3.3 Openvas Scanner

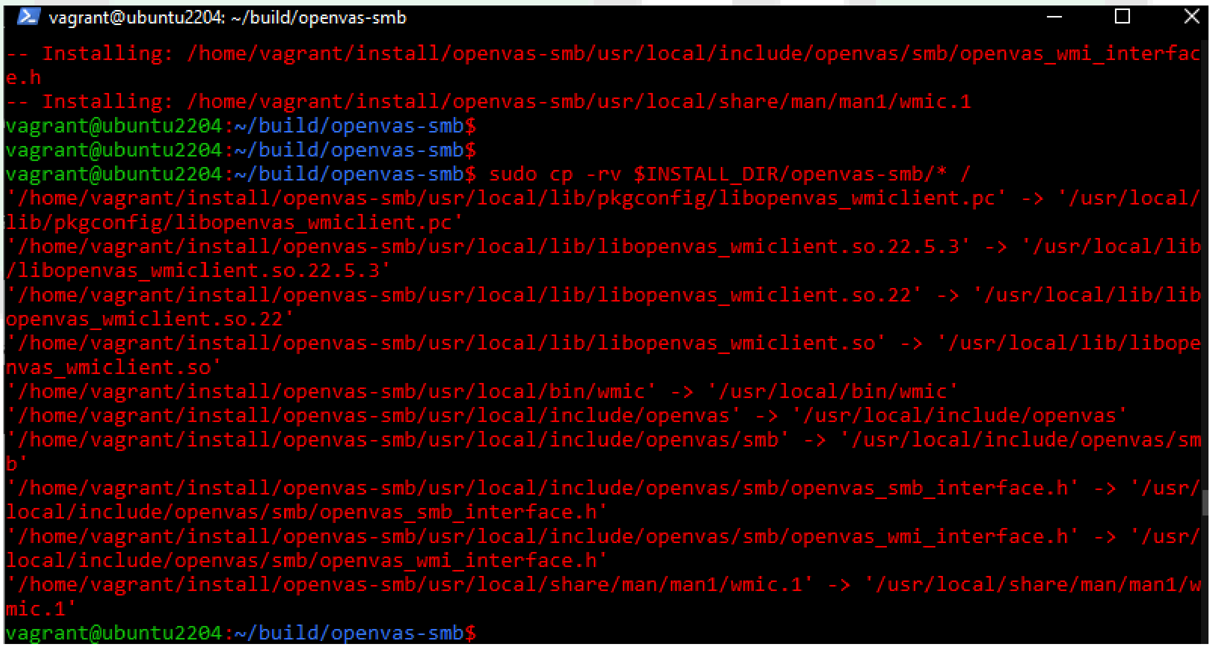
O openvas-scanner é o motor de varredura de vulnerabilidades do GVM, responsável por aplicar testes aos sistemas-alvo usando os scripts NVTs (Network Vulnerability Tests). Ele coleta informações dos alvos, identifica falhas de segurança e retorna os resultados para o gerenciador gvmd. Funciona em conjunto com o openvas-smb (para análise em ambientes Windows), com o ospd-openvas (interface de comunicação via OSP), ou com o novo openvasd para integração via API.

3.3.1 openvas-smb

Responsável pela autenticação SMB e análise de vulnerabilidades específicas em ambientes Windows.

Passo 6 - Prossiga copiando e colando o código abaixo no terminal.

```
tar -C "$SOURCE_DIR" -xvzf
"$SOURCE_DIR/openvas-smb-$OPENVAS_SMB_VERSION.tar.gz" && mkdir -p
"$BUILD_DIR/openvas-smb" && cmake -S
"$SOURCE_DIR/openvas-smb-$OPENVAS_SMB_VERSION" -B "$BUILD_DIR/openvas-smb"
-DCMAKE_INSTALL_PREFIX="$INSTALL_PREFIX" -DCMAKE_BUILD_TYPE=Release && cmake
--build "$BUILD_DIR/openvas-smb" -j"$(nproc)" && mkdir -p
"$INSTALL_DIR/openvas-smb" && cd "$BUILD_DIR/openvas-smb" && make
DESTDIR="$INSTALL_DIR/openvas-smb" install && sudo cp -rv
"$INSTALL_DIR/openvas-smb"/* /
```



```
vagrant@ubuntu2204: ~/build/openvas-smb
-- Installing: /home/vagrant/install/openvas-smb/usr/local/include/openvas/smb/openvas_wmi_interfac
e.h
-- Installing: /home/vagrant/install/openvas-smb/usr/local/share/man/man1/wmic.1
vagrant@ubuntu2204:~/build/openvas-smb$
vagrant@ubuntu2204:~/build/openvas-smb$
vagrant@ubuntu2204:~/build/openvas-smb$ sudo cp -rv $INSTALL_DIR/openvas-smb/* /
'/home/vagrant/install/openvas-smb/usr/local/lib/pkgconfig/libopenvas_wmiclient.pc' -> '/usr/local/
lib/pkgconfig/libopenvas_wmiclient.pc'
'/home/vagrant/install/openvas-smb/usr/local/lib/libopenvas_wmiclient.so.22.5.3' -> '/usr/local/lib
/libopenvas_wmiclient.so.22.5.3'
'/home/vagrant/install/openvas-smb/usr/local/lib/libopenvas_wmiclient.so.22' -> '/usr/local/lib/lib
openvas_wmiclient.so.22'
'/home/vagrant/install/openvas-smb/usr/local/lib/libopenvas_wmiclient.so' -> '/usr/local/lib/libope
nvas_wmiclient.so'
'/home/vagrant/install/openvas-smb/usr/local/bin/wmic' -> '/usr/local/bin/wmic'
'/home/vagrant/install/openvas-smb/usr/local/include/openvas' -> '/usr/local/include/openvas'
'/home/vagrant/install/openvas-smb/usr/local/include/openvas/smb' -> '/usr/local/include/openvas/sm
b'
'/home/vagrant/install/openvas-smb/usr/local/include/openvas/smb/openvas_smb_interface.h' -> '/usr/
local/include/openvas/smb/openvas_smb_interface.h'
'/home/vagrant/install/openvas-smb/usr/local/include/openvas/smb/openvas_wmi_interface.h' -> '/usr/
local/include/openvas/smb/openvas_wmi_interface.h'
'/home/vagrant/install/openvas-smb/usr/local/share/man/man1/wmic.1' -> '/usr/local/share/man/man1/w
mic.1'
vagrant@ubuntu2204:~/build/openvas-smb$
```

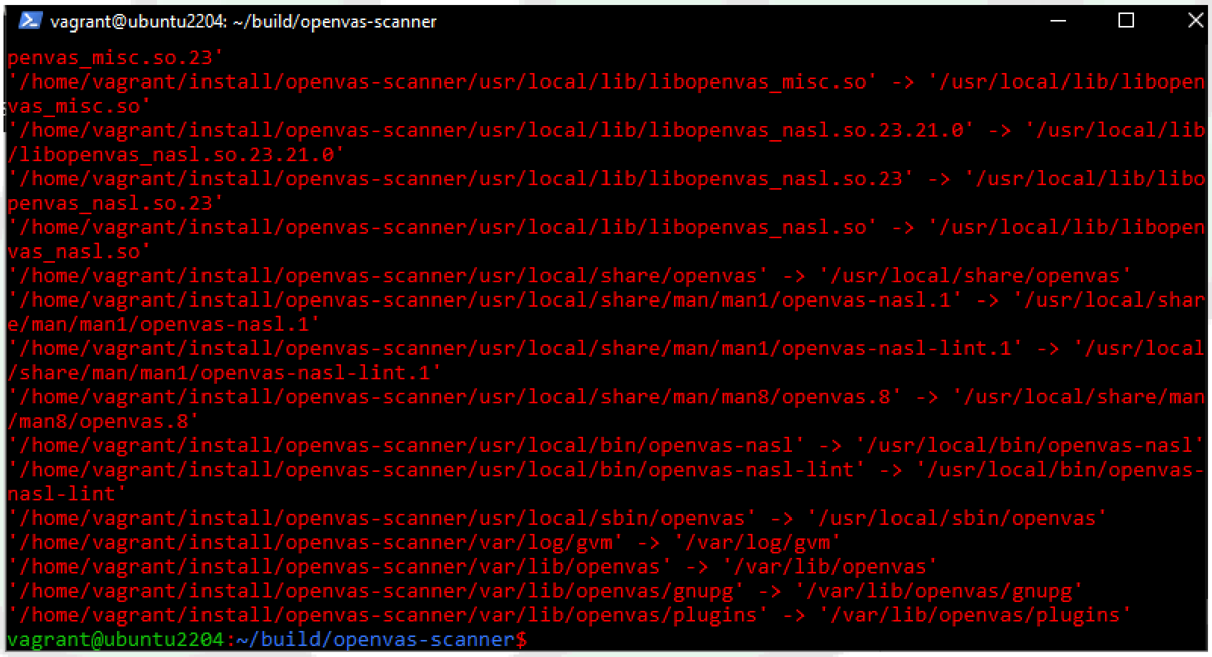
Figura 15: Instalação e transferência do openvas-smb para pasta padrão do sistema.

3.3.2 openvas-scanner

Módulo principal de varredura; executa os testes de segurança nos alvos definidos.

Passo 7 - Copie e cole todo o comando abaixo no terminal.

```
tar -C "$SOURCE_DIR" -xvzf
"$SOURCE_DIR/openvas-scanner-$OPENVAS_SCANNER_VERSION.tar.gz" && mkdir -p
"$BUILD_DIR/openvas-scanner" && cmake -S
"$SOURCE_DIR/openvas-scanner-$OPENVAS_SCANNER_VERSION" -B
"$BUILD_DIR/openvas-scanner" -DCMAKE_INSTALL_PREFIX="$INSTALL_PREFIX"
-DCMAKE_BUILD_TYPE=Release -DSYSCONFDIR=/etc -DLOCALSTATEDIR=/var
-DOPENVAS_FEED_LOCK_PATH=/var/lib/openvas/feed-update.lock
-DOPENVAS_RUN_DIR=/run/ospd && cmake --build "$BUILD_DIR/openvas-scanner"
-j"$(nproc)" && mkdir -p "$INSTALL_DIR/openvas-scanner" && cd
"$BUILD_DIR/openvas-scanner" && make DESTDIR="$INSTALL_DIR/openvas-scanner"
install && sudo cp -rv "$INSTALL_DIR/openvas-scanner"/* /
```

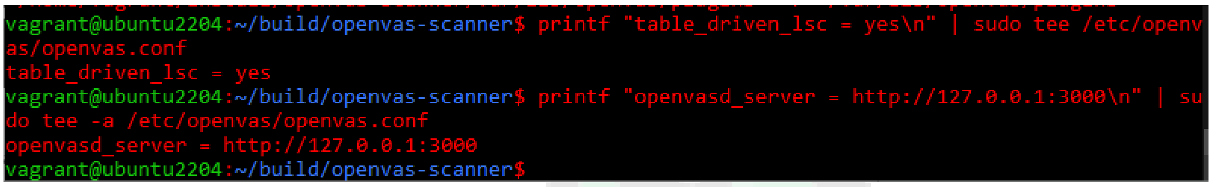


```
vagrant@ubuntu2204: ~/build/openvas-scanner
penvas_misc.so.23'
'/home/vagrant/install/openvas-scanner/usr/local/lib/libopenvas_misc.so' -> '/usr/local/lib/libopen
vas_misc.so'
'/home/vagrant/install/openvas-scanner/usr/local/lib/libopenvas_nasl.so.23.21.0' -> '/usr/local/lib
/libopenvas_nasl.so.23.21.0'
'/home/vagrant/install/openvas-scanner/usr/local/lib/libopenvas_nasl.so.23' -> '/usr/local/lib/libo
penvas_nasl.so.23'
'/home/vagrant/install/openvas-scanner/usr/local/lib/libopenvas_nasl.so' -> '/usr/local/lib/libopen
vas_nasl.so'
'/home/vagrant/install/openvas-scanner/usr/local/share/openvas' -> '/usr/local/share/openvas'
'/home/vagrant/install/openvas-scanner/usr/local/share/man/man1/openvas-nasl.1' -> '/usr/local/shar
e/man/man1/openvas-nasl.1'
'/home/vagrant/install/openvas-scanner/usr/local/share/man/man1/openvas-nasl-lint.1' -> '/usr/local
/share/man/man1/openvas-nasl-lint.1'
'/home/vagrant/install/openvas-scanner/usr/local/share/man/man8/openvas.8' -> '/usr/local/share/man
/man8/openvas.8'
'/home/vagrant/install/openvas-scanner/usr/local/bin/openvas-nasl' -> '/usr/local/bin/openvas-nasl'
'/home/vagrant/install/openvas-scanner/usr/local/bin/openvas-nasl-lint' -> '/usr/local/bin/openvas-
nasl-lint'
'/home/vagrant/install/openvas-scanner/usr/local/sbin/openvas' -> '/usr/local/sbin/openvas'
'/home/vagrant/install/openvas-scanner/var/log/gvm' -> '/var/log/gvm'
'/home/vagrant/install/openvas-scanner/var/lib/openvas' -> '/var/lib/openvas'
'/home/vagrant/install/openvas-scanner/var/lib/openvas/gnupg' -> '/var/lib/openvas/gnupg'
'/home/vagrant/install/openvas-scanner/var/lib/openvas/plugins' -> '/var/lib/openvas/plugins'
vagrant@ubuntu2204:~/build/openvas-scanner$
```

Figura 16: Instalação e transferência do openvas-scanner para pasta padrão do sistema.

Passo 8 - Ao concluir este processo de build, insira estes comandos para ajustar as variáveis de configuração do openvas-scanner.

```
printf "table_driven_lsc = yes\n" | sudo tee /etc/openvas/openvas.conf

printf "openvasd_server = http://127.0.0.1:3000\n" | sudo tee -a
/etc/openvas/openvas.conf
```



```
vagrant@ubuntu2204:~/build/openvas-scanner$ printf "table_driven_lsc = yes\n" | sudo tee /etc/openv
as/openvas.conf
table_driven_lsc = yes
vagrant@ubuntu2204:~/build/openvas-scanner$ printf "openvasd_server = http://127.0.0.1:3000\n" | su
do tee -a /etc/openvas/openvas.conf
openvasd_server = http://127.0.0.1:3000
vagrant@ubuntu2204:~/build/openvas-scanner$
```

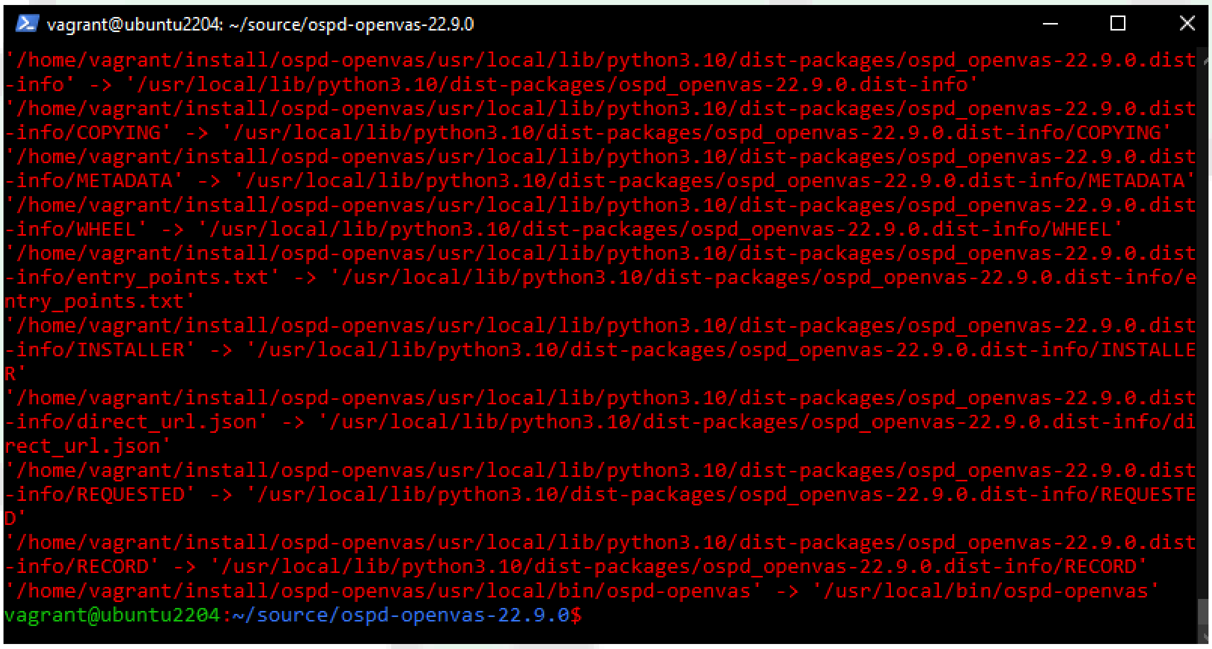
Figura 17: *Gerando arquivos de configuração do openvas-scanner.*

3.3.3 ospd-openvas

Camada de comunicação que conecta o scanner ao gvmd via protocolo OSP.

Passo 9 - Copie e cole o código e aguarde a finalização.

```
tar -C "$SOURCE_DIR" -xvzf
"$SOURCE_DIR/ospd-openvas-$OSPD_OPENVAS_VERSION.tar.gz" && cd
"$SOURCE_DIR/ospd-openvas-$OSPD_OPENVAS_VERSION" && mkdir -p
"$INSTALL_DIR/ospd-openvas" && python3 -m pip install
--root="$INSTALL_DIR/ospd-openvas" --no-warn-script-location . && sudo cp -rv
"$INSTALL_DIR/ospd-openvas"/* /
```



vagrant@ubuntu2204: ~/source/ospd-openvas-22.9.0

```
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/COPYING' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/COPYING'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/METADATA' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/METADATA'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/WHEEL' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/WHEEL'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/entry_points.txt' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/e
ntry_points.txt'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/INSTALLER' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/INSTALLE
R'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/direct_url.json' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/di
rect_url.json'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/REQUESTED' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/REQUESTE
D'
'/home/vagrant/install/ospd-openvas/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist
-info/RECORD' -> '/usr/local/lib/python3.10/dist-packages/ospd_openvas-22.9.0.dist-info/RECORD'
'/home/vagrant/install/ospd-openvas/usr/local/bin/ospd-openvas' -> '/usr/local/bin/ospd-openvas'
vagrant@ubuntu2204:~/source/ospd-openvas-22.9.0$
```

Figura 18: *Instalação e transferência do ospd-openvas para pasta padrão do sistema.*

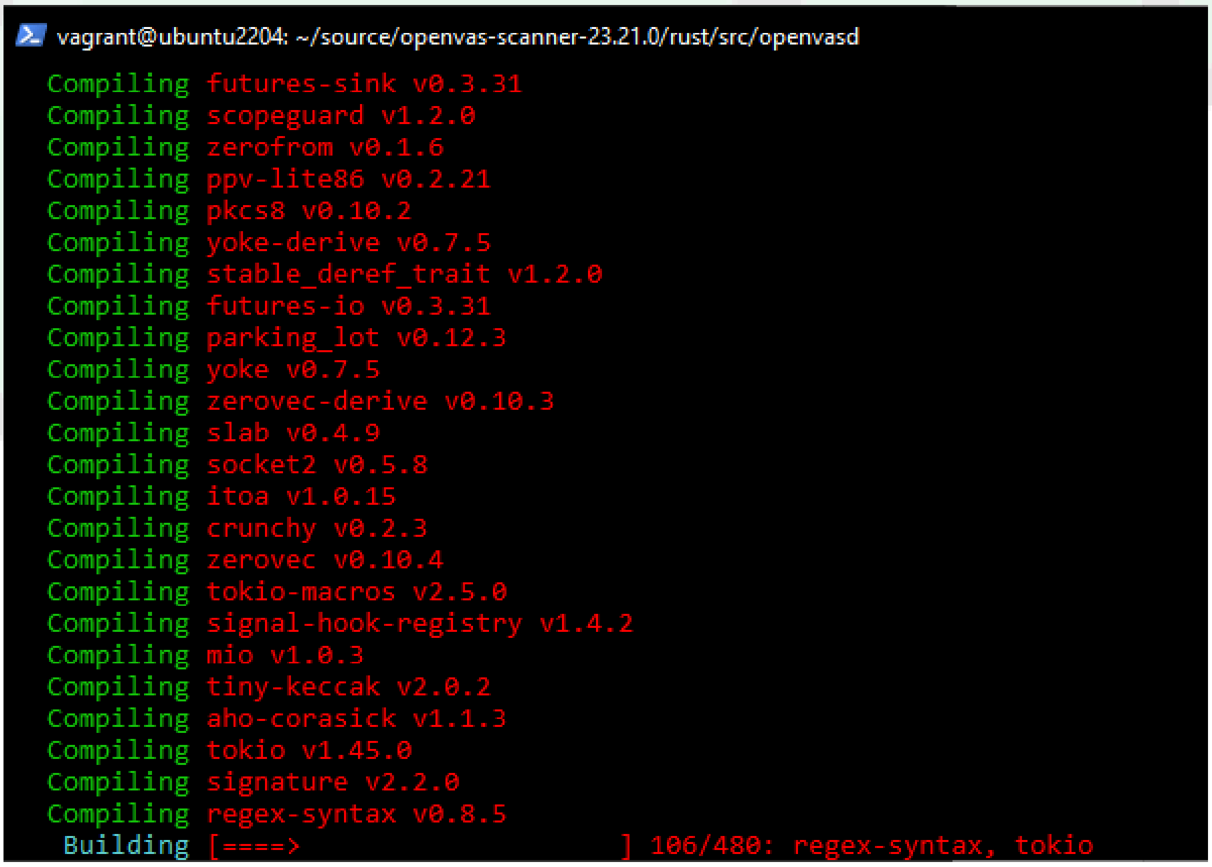
3.3.4 openvasd

Nova versão em Rust que integra o Notus Scanner e substituirá o ospd-openvas, oferecendo escaneamento via API RESTful.

Passo 10 - Copie e cole o código a seguir no terminal, o processo de instalação deste módulo pode ser acompanhado nas imagens abaixo.

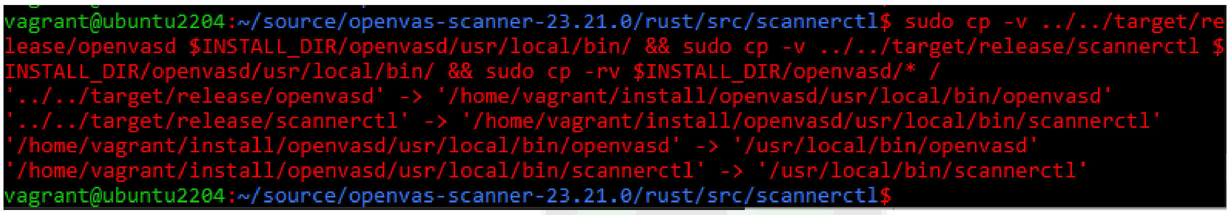
⚠️ Este processo costuma demorar um pouco, então reserve um tempo.

```
tar -C "$SOURCE_DIR" -xvzf
"$SOURCE_DIR/openvas-scanner-$OPENVAS_DAEMON.tar.gz" && mkdir -p
"$INSTALL_DIR/openvasd/usr/local/bin" && cd
"$SOURCE_DIR/openvas-scanner-$OPENVAS_DAEMON/rust/src/openvasd" && cargo build
--release && cd
"$SOURCE_DIR/openvas-scanner-$OPENVAS_DAEMON/rust/src/scannerctl" && cargo
build --release && sudo cp -v ../../target/release/openvasd
"$INSTALL_DIR/openvasd/usr/local/bin/" && sudo cp -v
../../target/release/scannerctl "$INSTALL_DIR/openvasd/usr/local/bin/" && sudo
cp -rv "$INSTALL_DIR/openvasd"/* /
```



```
vagrant@ubuntu2204: ~/source/openvas-scanner-23.21.0/rust/src/openvasd
   Compiling futures-sink v0.3.31
   Compiling scopeguard v1.2.0
   Compiling zerofrom v0.1.6
   Compiling ppv-lite86 v0.2.21
   Compiling pkcs8 v0.10.2
   Compiling yoke-derive v0.7.5
   Compiling stable_deref_trait v1.2.0
   Compiling futures-io v0.3.31
   Compiling parking_lot v0.12.3
   Compiling yoke v0.7.5
   Compiling zerovec-derive v0.10.3
   Compiling slab v0.4.9
   Compiling socket2 v0.5.8
   Compiling itoa v1.0.15
   Compiling crunchy v0.2.3
   Compiling zerovec v0.10.4
   Compiling tokio-macros v2.5.0
   Compiling signal-hook-registry v1.4.2
   Compiling mio v1.0.3
   Compiling tiny-keccak v2.0.2
   Compiling aho-corasick v1.1.3
   Compiling tokio v1.45.0
   Compiling signature v2.2.0
   Compiling regex-syntax v0.8.5
    Building [====>                        ] 106/480: regex-syntax, tokio
```

Figura 19: Compilação e instalação dos arquivos do openvasd através do cargo (rust).



```
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$ sudo cp -v ../../target/re
lease/openvasd $INSTALL_DIR/openvasd/usr/local/bin/ && sudo cp -v ../../target/release/scannerctl $
INSTALL_DIR/openvasd/usr/local/bin/ && sudo cp -rv $INSTALL_DIR/openvasd/* /
'../../target/release/openvasd' -> '/home/vagrant/install/openvasd/usr/local/bin/openvasd'
'../../target/release/scannerctl' -> '/home/vagrant/install/openvasd/usr/local/bin/scannerctl'
'/home/vagrant/install/openvasd/usr/local/bin/openvasd' -> '/usr/local/bin/openvasd'
'/home/vagrant/install/openvasd/usr/local/bin/scannerctl' -> '/usr/local/bin/scannerctl'
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$
```

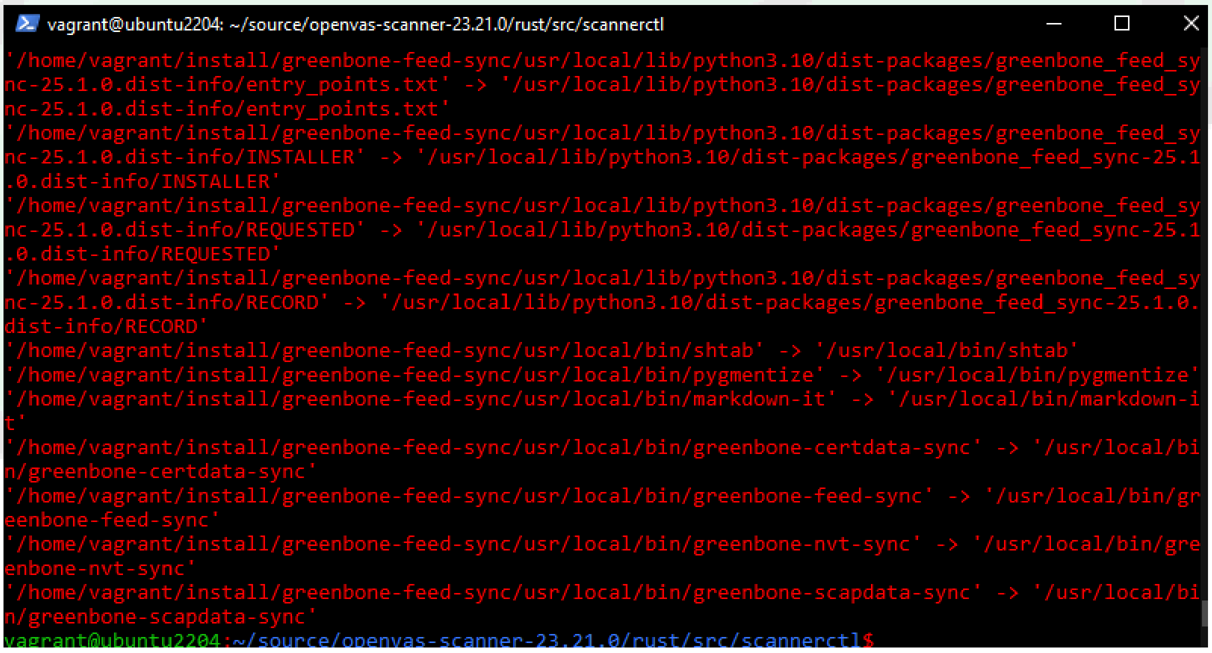
Figura 20: Transferência do openvasd para pasta padrão do sistema.

3.4 greenbone-feed-sync

O greenbone-feed-sync é um utilitário responsável por sincronizar os Feeds de vulnerabilidades usados pelo GVM. Ele baixa e atualiza os dados de testes (NVTs), CERTs e CPEs diretamente do servidor da Greenbone. Isso garante que o escaneamento esteja sempre com a base mais recente de detecção de falhas.

Passo 11 - Copie e cole o código abaixo, a instalação será realizada via pip.

```
mkdir -p "$INSTALL_DIR/greenbone-feed-sync" && python3 -m pip install
--root="$INSTALL_DIR/greenbone-feed-sync" --no-warn-script-location
greenbone-feed-sync && sudo cp -rv "$INSTALL_DIR/greenbone-feed-sync"/* /
```



```
vagrant@ubuntu2204: ~/source/openvas-scanner-23.21.0/rust/src/scannerctl
'/home/vagrant/install/greenbone-feed-sync/usr/local/lib/python3.10/dist-packages/greenbone_feed_sy
nc-25.1.0.dist-info/entry_points.txt' -> '/usr/local/lib/python3.10/dist-packages/greenbone_feed_sy
nc-25.1.0.dist-info/entry_points.txt'
'/home/vagrant/install/greenbone-feed-sync/usr/local/lib/python3.10/dist-packages/greenbone_feed_sy
nc-25.1.0.dist-info/INSTALLER' -> '/usr/local/lib/python3.10/dist-packages/greenbone_feed_sync-25.1
.0.dist-info/INSTALLER'
'/home/vagrant/install/greenbone-feed-sync/usr/local/lib/python3.10/dist-packages/greenbone_feed_sy
nc-25.1.0.dist-info/REQUESTED' -> '/usr/local/lib/python3.10/dist-packages/greenbone_feed_sync-25.1
.0.dist-info/REQUESTED'
'/home/vagrant/install/greenbone-feed-sync/usr/local/lib/python3.10/dist-packages/greenbone_feed_sy
nc-25.1.0.dist-info/RECORD' -> '/usr/local/lib/python3.10/dist-packages/greenbone_feed_sync-25.1.0.
dist-info/RECORD'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/shtab' -> '/usr/local/bin/shtab'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/pygmentize' -> '/usr/local/bin/pygmentize'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/markdown-it' -> '/usr/local/bin/markdown-i
t'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/greenbone-certdata-sync' -> '/usr/local/bi
n/greenbone-certdata-sync'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/greenbone-feed-sync' -> '/usr/local/bin/gr
eenbone-feed-sync'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/greenbone-nvt-sync' -> '/usr/local/bin/gre
enbone-nvt-sync'
'/home/vagrant/install/greenbone-feed-sync/usr/local/bin/greenbone-scapdata-sync' -> '/usr/local/bi
n/greenbone-scapdata-sync'
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$
```

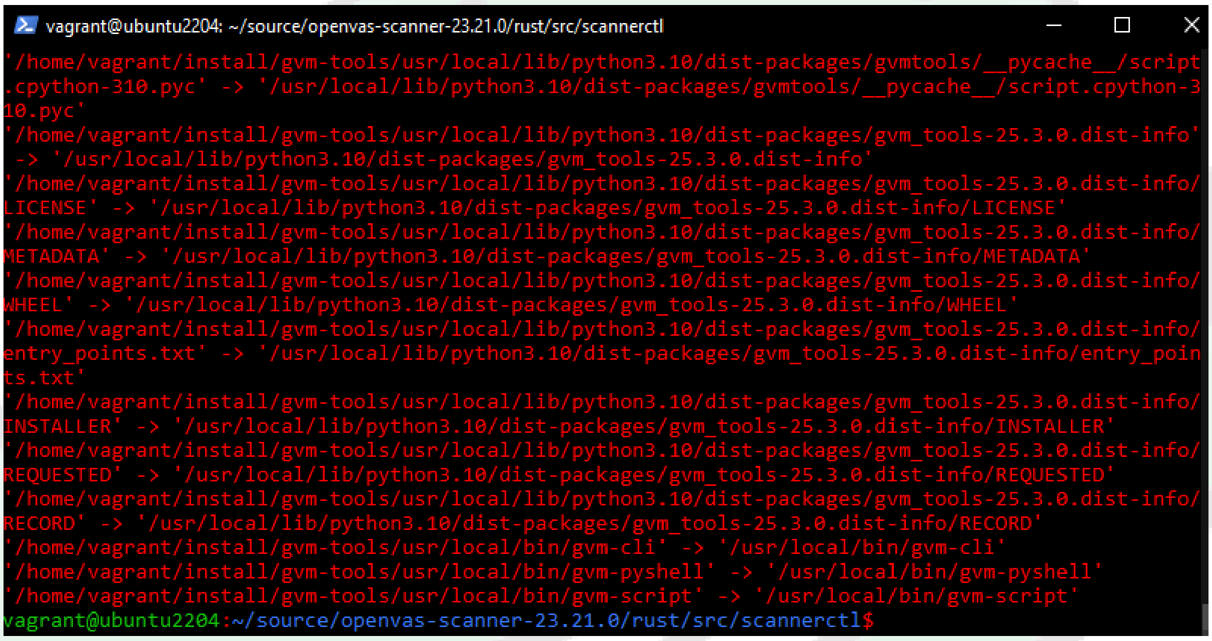
Figura 21: Instalação e transferência do greenbone-feed-sync para pasta padrão do sistema.

3.5 gvm-tools

O gvm-tools é um conjunto de scripts e utilitários para interagir com o gerenciador gvmd via API (GMP ou OSP). Permite automatizar tarefas como criação de scans, importação de targets e extração de relatórios sem usar a interface web.

Passo 12 - Cole o código abaixo no terminal e aguarde o término.

```
mkdir -p "$INSTALL_DIR/gvm-tools" && python3 -m pip install
--root="$INSTALL_DIR/gvm-tools" --no-warn-script-location gvm-tools && sudo cp
-rv "$INSTALL_DIR/gvm-tools"/* /
```



```
vagrant@ubuntu2204: ~/source/openvas-scanner-23.21.0/rust/src/scannerctl
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvmtools/__pycache__/script
.cpython-310.pyc' -> '/usr/local/lib/python3.10/dist-packages/gvmtools/__pycache__/script.cpython-3
10.pyc'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info'
 -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
LICENSE' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/LICENSE'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
METADATA' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/METADATA'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
WHEEL' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/WHEEL'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
entry_points.txt' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/entry_poin
ts.txt'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
INSTALLER' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/INSTALLER'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
REQUESTED' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/REQUESTED'
'/home/vagrant/install/gvm-tools/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/
RECORD' -> '/usr/local/lib/python3.10/dist-packages/gvm_tools-25.3.0.dist-info/RECORD'
'/home/vagrant/install/gvm-tools/usr/local/bin/gvm-cli' -> '/usr/local/bin/gvm-cli'
'/home/vagrant/install/gvm-tools/usr/local/bin/gvm-pyshell' -> '/usr/local/bin/gvm-pyshell'
'/home/vagrant/install/gvm-tools/usr/local/bin/gvm-script' -> '/usr/local/bin/gvm-script'
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$
```

Figura 22: Instalação e transferência do gvm-tools para pasta padrão do sistema.

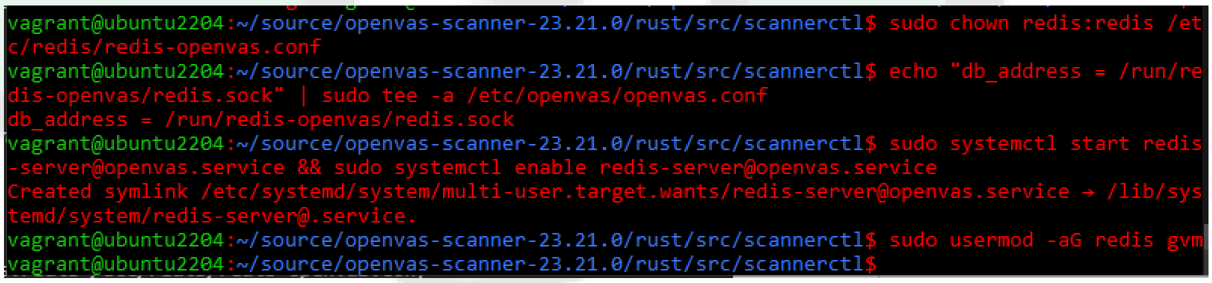
4. Instalações adicionais e ajuste de configurações

4.1 redis-server

O redis-server é utilizado como banco de dados temporário de alto desempenho para armazenar dados em cache durante os scans do GVM. Ele melhora a velocidade e a eficiência das operações, especialmente nas interações entre os módulos como gvmd e openvas-scanner.

Passo 1 - Realize a instalação e a configuração do redis-server, depois crie um serviço para uso do Greenbone através dos comandos abaixo.

```
sudo apt install -y redis-server

sudo cp
$SOURCE_DIR/openvas-scanner-$OPENVAS_SCANNER_VERSION/config/redis-openvas.conf
/etc/redis/

sudo chown redis:redis /etc/redis/redis-openvas.conf

echo "db_address = /run/redis-openvas/redis.sock" | sudo tee -a
/etc/openvas/openvas.conf

sudo systemctl start redis-server@openvas.service && sudo systemctl enable
redis-server@openvas.service

sudo usermod -aG redis gvm
```



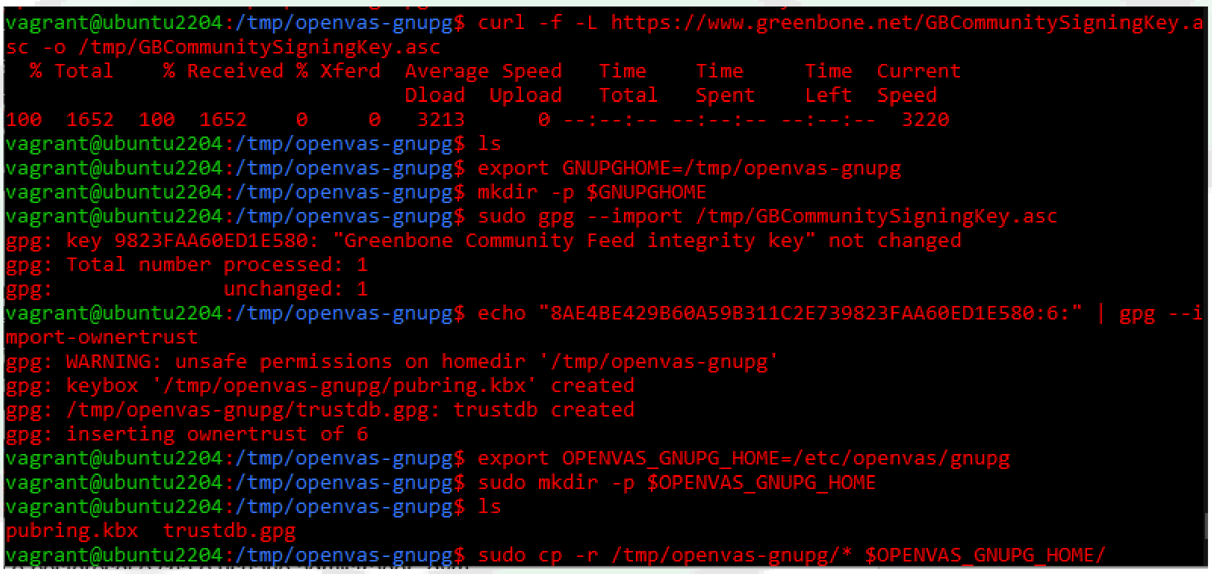
```
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$ sudo chown redis:redis /et
c/redis/redis-openvas.conf
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$ echo "db_address = /run/re
dis-openvas/redis.sock" | sudo tee -a /etc/openvas/openvas.conf
db_address = /run/redis-openvas/redis.sock
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$ sudo systemctl start redis
-server@openvas.service && sudo systemctl enable redis-server@openvas.service
Created symlink /etc/systemd/system/multi-user.target.wants/redis-server@openvas.service → /lib/sys
temd/system/redis-server@.service.
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$ sudo usermod -aG redis gvm
vagrant@ubuntu2204:~/source/openvas-scanner-23.21.0/rust/src/scannerctl$
```

Figura 23: Configuração e inicialização do serviço redis-server@openvas e atribuição do usuário gvm ao grupo redis.

4.2 Ajustes de Configurações e Permissões

Passo 2 - Ajuste as permissões e importe a chave para a validação dos Feeds.

```
sudo mkdir -p /var/lib/notus /run/gvmd && sudo chown -R gvm:gvm
/var/lib/{gvm,openvas,notus} /var/log/gvm /run/gvmd && sudo chmod -R g+srw
/var/lib/{gvm,openvas} /var/log/gvm && sudo chown gvm:gvm /usr/local/sbin/gvmd
&& sudo chmod 6750 /usr/local/sbin/gvmd

mkdir -p $GNUPGHOME

sudo gpg --import /tmp/GBCommunitySigningKey.asc

echo "8AE4BE429B60A59B311C2E739823FAA60ED1E580:6:" | gpg --import-ownertrust

sudo mkdir -p $OPENVAS_GNUPG_HOME

sudo cp -r /tmp/openvas-gnupg/* $OPENVAS_GNUPG_HOME/

sudo chown -R gvm:gvm $OPENVAS_GNUPG_HOME
```

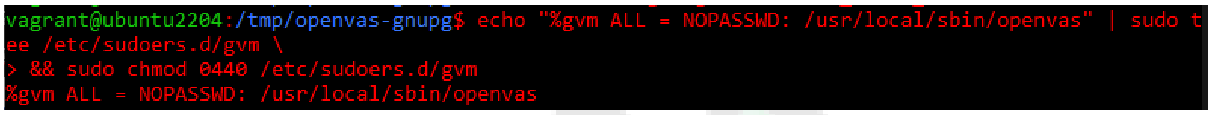


```
vagrant@ubuntu2204:/tmp/openvas-gnupg$ curl -f -L https://www.greenbone.net/GBCommunitySigningKey.a
sc -o /tmp/GBCommunitySigningKey.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1652  100  1652    0     0   3213      0 --:--:-- --:--:-- --:--:--  3220
vagrant@ubuntu2204:/tmp/openvas-gnupg$ ls
vagrant@ubuntu2204:/tmp/openvas-gnupg$ export GNUPGHOME=/tmp/openvas-gnupg
vagrant@ubuntu2204:/tmp/openvas-gnupg$ mkdir -p $GNUPGHOME
vagrant@ubuntu2204:/tmp/openvas-gnupg$ sudo gpg --import /tmp/GBCommunitySigningKey.asc
gpg: key 9823FAA60ED1E580: "Greenbone Community Feed integrity key" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
vagrant@ubuntu2204:/tmp/openvas-gnupg$ echo "8AE4BE429B60A59B311C2E739823FAA60ED1E580:6:" | gpg --i
mport-ownertrust
gpg: WARNING: unsafe permissions on homedir '/tmp/openvas-gnupg'
gpg: keybox '/tmp/openvas-gnupg/pubring.kbx' created
gpg: /tmp/openvas-gnupg/trustdb.gpg: trustdb created
gpg: inserting ownertrust of 6
vagrant@ubuntu2204:/tmp/openvas-gnupg$ export OPENVAS_GNUPG_HOME=/etc/openvas/gnupg
vagrant@ubuntu2204:/tmp/openvas-gnupg$ sudo mkdir -p $OPENVAS_GNUPG_HOME
vagrant@ubuntu2204:/tmp/openvas-gnupg$ ls
pubring.kbx  trustdb.gpg
vagrant@ubuntu2204:/tmp/openvas-gnupg$ sudo cp -r /tmp/openvas-gnupg/* $OPENVAS_GNUPG_HOME/
```

Figura 24: Importando a chave de integridade para validação dos feeds.

Passo 3 - Como alguns módulos exigem privilégios administrativos, configure o sudo para permitir que o usuário gvm tenha acesso aos comandos necessários.

```
echo "%gvm ALL = NOPASSWD: /usr/local/sbin/openvas" | sudo tee
/etc/sudoers.d/gvm \

&& sudo chmod 0440 /etc/sudoers.d/gvm
```

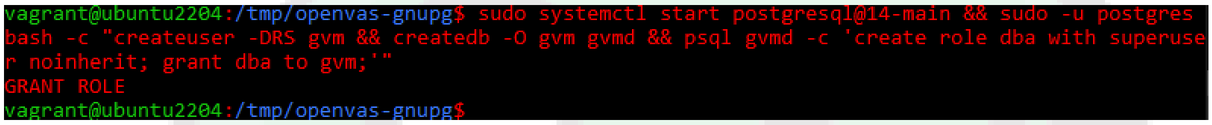


```
vagrant@ubuntu2204:/tmp/openvas-gnupg$ echo "%gvm ALL = NOPASSWD: /usr/local/sbin/openvas" | sudo t
ee /etc/sudoers.d/gvm \
> && sudo chmod 0440 /etc/sudoers.d/gvm
%gvm ALL = NOPASSWD: /usr/local/sbin/openvas
```

Figura 25: Atribuindo o sudo ao usuário gvm.

Passo 4 - Retorne à pasta home, inicie o serviço do PostgreSQL e crie o usuário gvm, atribuindo-lhe permissões administrativas para uso com o banco de dados.

```
cd

sudo systemctl start postgresql@14-main && sudo -u postgres bash -c
"createuser -DRS gvm && createdb -O gvm gvmd && psql gvmd -c 'create role dba
with superuser noinherit; grant dba to gvm;'"
```



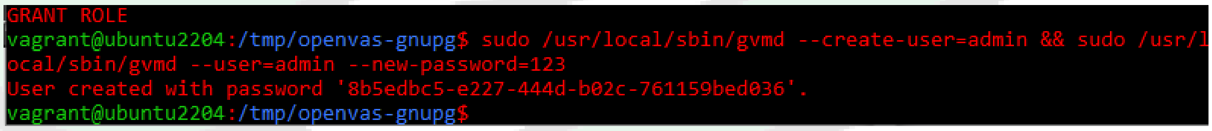
```
vagrant@ubuntu2204:/tmp/openvas-gnupg$ sudo systemctl start postgresql@14-main && sudo -u postgres
bash -c "createuser -DRS gvm && createdb -O gvm gvmd && psql gvmd -c 'create role dba with superuse
r noinherit; grant dba to gvm;'"
GRANT ROLE
vagrant@ubuntu2204:/tmp/openvas-gnupg$
```

Figura 26: Inicializando o processo do postgresql, criando usuário gvm e atribuindo privilégios.

4.3 Criação e configuração do usuário administrador

Passo 5 - Configure usuário admin, e em "--new-password" coloque uma senha de sua preferência, uma outra senha será impressa ao fim do comando, porém ela será logo sobrescrita pela senha que você definiu na variável.

```
sudo /usr/local/sbin/gvmd --create-user=admin && sudo /usr/local/sbin/gvmd
--user=admin --new-password=123
```

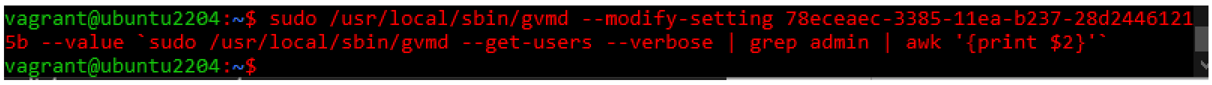


```
GRANT ROLE
vagrant@ubuntu2204:/tmp/openvas-gnupg$ sudo /usr/local/sbin/gvmd --create-user=admin && sudo /usr/l
ocal/sbin/gvmd --user=admin --new-password=123
User created with password '8b5edbc5-e227-444d-b02c-761159bed036'.
vagrant@ubuntu2204:/tmp/openvas-gnupg$
```

Figura 27: Criando usuário admin e atribuindo uma nova senha.

Passo 6 - Ajuste a permissão para que os feeds sejam gerenciados pelo usuário "Admin" do Greenbone.

```
sudo /usr/local/sbin/gvmd --modify-setting
78eceaec-3385-11ea-b237-28d24461215b --value `sudo /usr/local/sbin/gvmd
--get-users --verbose | grep admin | awk '{print $2}'`
```



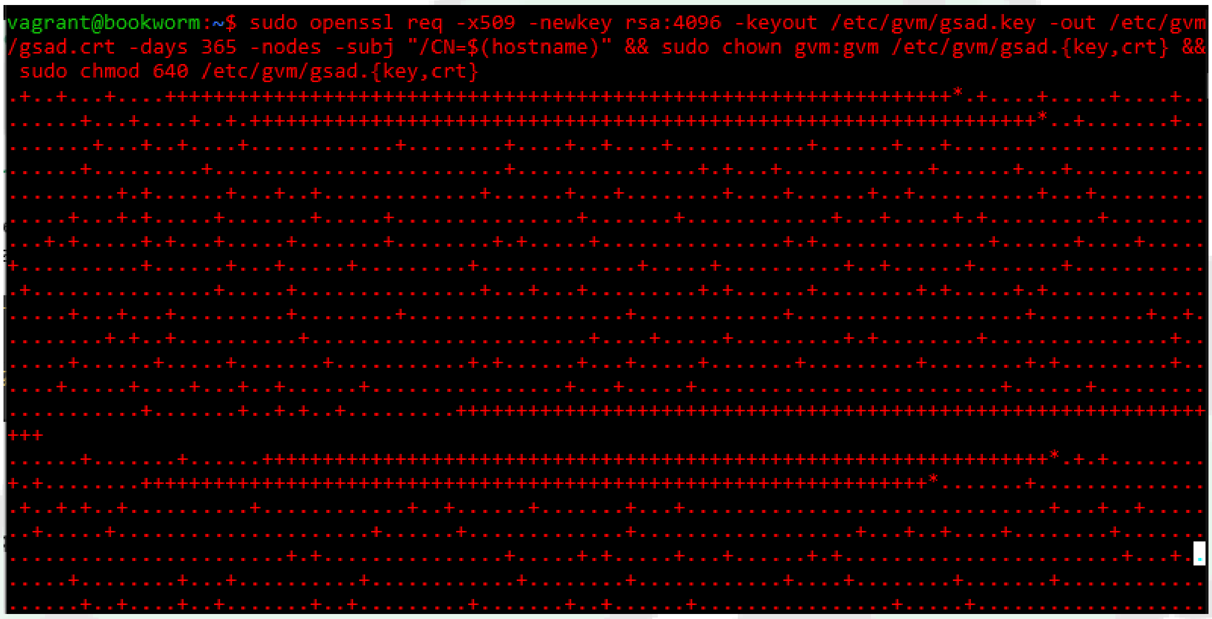
```
vagrant@ubuntu2204:~$ sudo /usr/local/sbin/gvmd --modify-setting 78eceaec-3385-11ea-b237-28d2446121
5b --value `sudo /usr/local/sbin/gvmd --get-users --verbose | grep admin | awk '{print $2}'`
vagrant@ubuntu2204:~$
```

Figura 28: Ajustando administração dos feeds.

4.4 Gerando certificado para acesso via HTTPS

Passo 7 - Gere um certificado auto nomeado para configurar a conexão via HTTPS e mude sua propriedade para o usuário gvm.

```
sudo openssl req -x509 -newkey rsa:4096 -keyout /etc/gvm/gsad.key -out
/etc/gvm/gsad.crt -days 365 -nodes -subj "/CN=$(hostname)" && sudo chown
gvm:gvm /etc/gvm/gsad.{key,crt} && sudo chmod 640 /etc/gvm/gsad.{key,crt}
```



```
vagrant@bookworm:~$ sudo openssl req -x509 -newkey rsa:4096 -keyout /etc/gvm/gsad.key -out /etc/gvm
/gsad.crt -days 365 -nodes -subj "/CN=$(hostname)" && sudo chown gvm:gvm /etc/gvm/gsad.{key,crt} &&
 sudo chmod 640 /etc/gvm/gsad.{key,crt}
.+..+...+....+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.+....+.....+....+..
......+...+....+..+.+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..+.......+..
.......+...+..+....+............+........+....+..+....+...........+......+...+...................
......+.........+........................+...............+.+...+............+......+...+.........
.........+.+......+...+..+............+......+...+........+....+......+..+...........+...+.......
.....+...+.+.....+.......+.....+................+.......+............+...+.....+.+..........+.......
...+.+.....+.+...+.....+.......+.........+.+.....+...............+.+...........+......+....+.....
+..........+......+...+.....+.........+..............+.....+..........+..+......+.......+..........
.+..............+.....+...............+...+...+.........+.+.......+.........+.+.....+.+..........
.....+...+...+.........+........+...................+............+...................+.........+..+.
........+.+..+..........+.........................+....+.....+.........+.+........+................+..
.....+......+.....+.......+...........+.+.......+...+.....+.......+..........+.........+.+.........+..
....+.....+....+...+..+......+................+...+......+..........................+.......+.........
...........+........+..+.+..+.........+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++
......+.......+.......+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.+.+........
+.+.........++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.......+.............
.+..+.+..+...........+............+..+.......+.......+...+.........................+....+..+.....
..+.....+.......................+......+.............+.....................+...+..+....+........+.......
.........................+.+...............+.....+.+.....+...+.......+.+.........................+...+.
.....+........+...+...........+.............+........+..........+....+.........+.......+..........
......+..+.....+..+.......+..+.........+.......+..+......+.......................+.....+...........
```

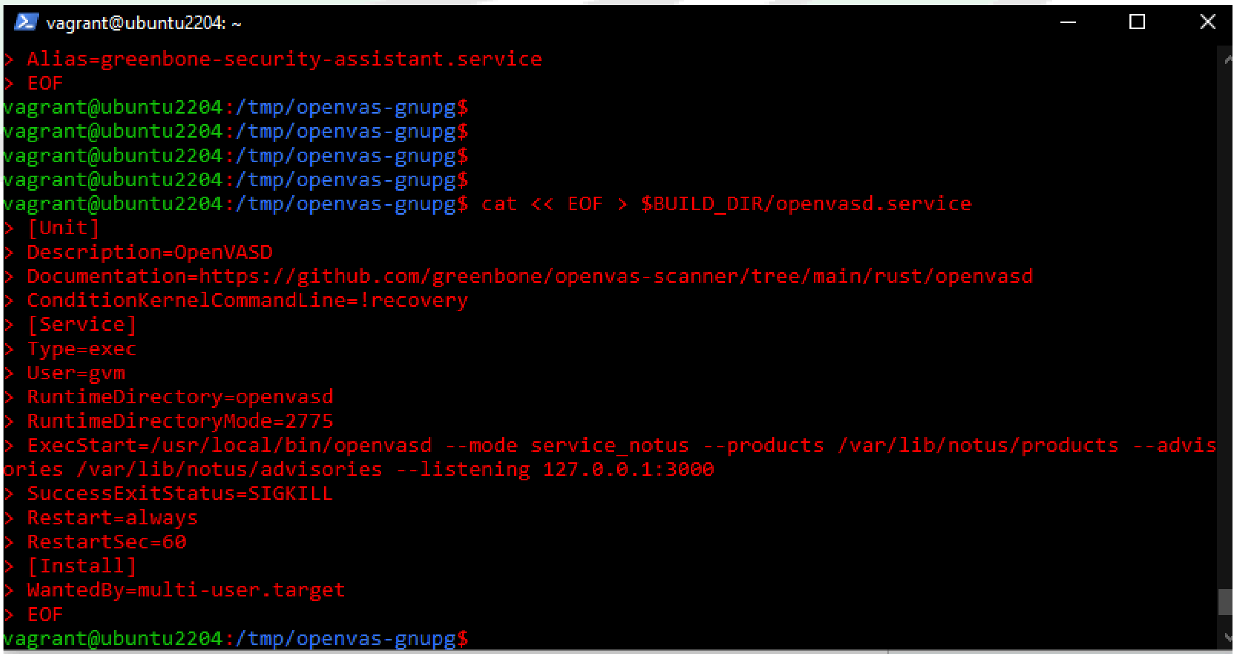
Figura 29: Gerando certificados para utilizar no HTTPS.

4.5 Atribuindo o Serviço ao Systemd

Passo 8 - Para gerar os arquivos referentes aos serviços de cada módulo do Greenbone, copie e cole todo o código abaixo.

```
cat << EOF > $BUILD_DIR/ospd-openvas.service
[Unit]
Description=OSPd Wrapper for the OpenVAS Scanner (ospd-openvas)
Documentation=man:ospd-openvas(8) man:openvas(8)
After=network.target networking.service redis-server@openvas.service
openvasd.service
Wants=redis-server@openvas.service openvasd.service
ConditionKernelCommandLine=!recovery


[Service]
Type=exec
User=gvm
Group=gvm
RuntimeDirectory=ospd
RuntimeDirectoryMode=2775
```

```
PIDFile=/run/ospd/ospd-openvas.pid
ExecStart=/usr/local/bin/ospd-openvas --foreground --unix-socket
/run/ospd/ospd-openvas.sock --pid-file /run/ospd/ospd-openvas.pid --log-file
/var/log/gvm/ospd-openvas.log --lock-file-dir /var/lib/openvas --socket-mode
0o770 --notus-feed-dir /var/lib/notus/advisories
SuccessExitStatus=SIGKILL
Restart=always
RestartSec=60


[Install]
WantedBy=multi-user.target
EOF


cat << EOF > $BUILD_DIR/gvmd.service
[Unit]
Description=Greenbone Vulnerability Manager daemon (gvmd)
After=network.target networking.service postgresql.service
ospd-openvas.service
Wants=postgresql.service ospd-openvas.service
Documentation=man:gvmd(8)
ConditionKernelCommandLine=!recovery


[Service]
Type=exec
User=gvm
Group=gvm
PIDFile=/run/gvmd/gvmd.pid
RuntimeDirectory=gvmd
RuntimeDirectoryMode=2775
ExecStart=/usr/local/sbin/gvmd --foreground
--osp-vt-update=/run/ospd/ospd-openvas.sock --listen-group=gvm
Restart=always
TimeoutStopSec=10


[Install]
WantedBy=multi-user.target
EOF


cat << EOF > $BUILD_DIR/gsad.service
[Unit]
Description=Greenbone Security Assistant daemon (gsad)
Documentation=man:gsad(8) https://www.greenbone.net
After=network.target gvmd.service
Wants=gvmd.service


[Service]
Type=exec
User=gvm
Group=gvm
RuntimeDirectory=gsad
RuntimeDirectoryMode=2775
PIDFile=/run/gsad/gsad.pid
ExecStart=/usr/local/sbin/gsad --listen=0.0.0.0 --port=9392 \
  --ssl-private-key=/etc/gvm/gsad.key \
  --ssl-certificate=/etc/gvm/gsad.crt
Restart=always
TimeoutStopSec=10


[Install]
WantedBy=multi-user.target
Alias=greenbone-security-assistant.service
EOF



cat << EOF > $BUILD_DIR/openvasd.service
[Unit]
Description=OpenVASD
Documentation=https://github.com/greenbone/openvas-scanner/tree/main/rust/open
vasd
ConditionKernelCommandLine=!recovery


[Service]
Type=exec
User=gvm
RuntimeDirectory=openvasd
RuntimeDirectoryMode=2775
ExecStart=/usr/local/bin/openvasd --mode service_notus --products
/var/lib/notus/products --advisories /var/lib/notus/advisories --listening
127.0.0.1:3000
SuccessExitStatus=SIGKILL
Restart=always
RestartSec=60


[Install]
WantedBy=multi-user.target
EOF
```

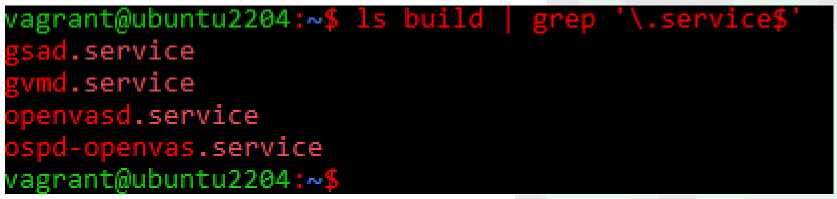


```
> Alias=greenbone-security-assistant.service
> EOF
vagrant@ubuntu2204:/tmp/openvas-gnupg$
vagrant@ubuntu2204:/tmp/openvas-gnupg$
vagrant@ubuntu2204:/tmp/openvas-gnupg$
vagrant@ubuntu2204:/tmp/openvas-gnupg$
vagrant@ubuntu2204:/tmp/openvas-gnupg$ cat << EOF > $BUILD_DIR/openvasd.service
> [Unit]
> Description=OpenVASD
> Documentation=https://github.com/greenbone/openvas-scanner/tree/main/rust/openvasd
> ConditionKernelCommandLine=!recovery
> [Service]
> Type=exec
> User=gvm
> RuntimeDirectory=openvasd
> RuntimeDirectoryMode=2775
> ExecStart=/usr/local/bin/openvasd --mode service_notus --products /var/lib/notus/products --advis
ories /var/lib/notus/advisories --listening 127.0.0.1:3000
> SuccessExitStatus=SIGKILL
> Restart=always
> RestartSec=60
> [Install]
> WantedBy=multi-user.target
> EOF
vagrant@ubuntu2204:/tmp/openvas-gnupg$
```

Figura 30: Criando arquivos dos serviços referentes aos módulos do Greenbone.

Passo 9 - Para confirmar se tudo foi criado corretamente, verifique o conteúdo da pasta build. Devem estar presentes quatro arquivos com extensão .service.

```
ls build | grep '\.service$'
```

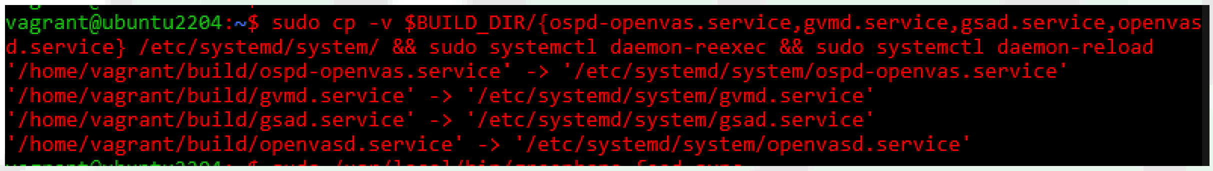


```
vagrant@ubuntu2204:~$ ls build | grep '\.service$'
gsad.service
gvmd.service
openvasd.service
ospd-openvas.service
vagrant@ubuntu2204:~$
```

Figura 31: Conferindo arquivos .service na pasta.

Passo 10 - Agora importe os serviços para o systemd, e reinicie o daemon.

```
sudo cp -v
$BUILD_DIR/{ospd-openvas.service,gvmd.service,gsad.service,openvasd.service}
/etc/systemd/system/ && sudo systemctl daemon-reexec && sudo systemctl
daemon-reload
```



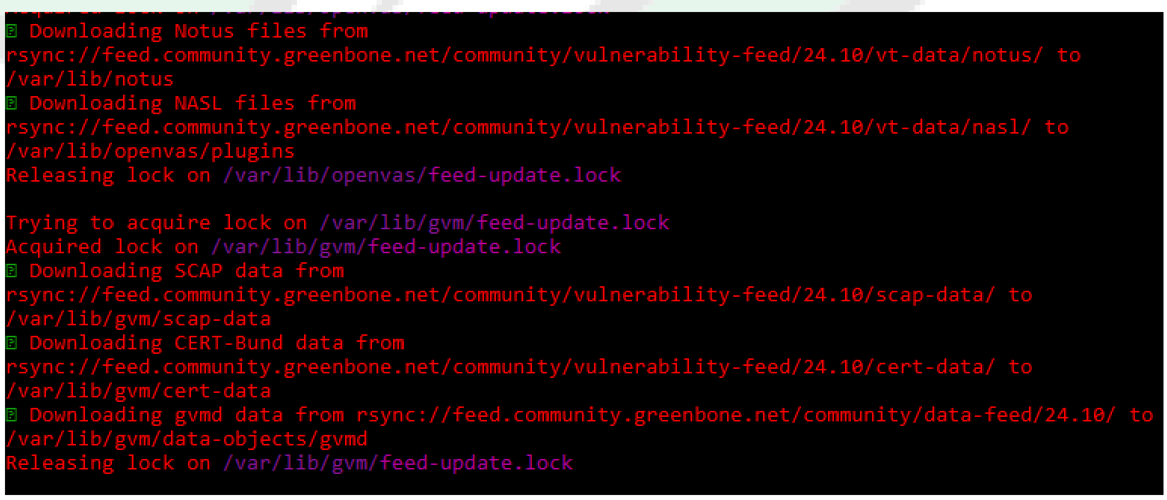
```
vagrant@ubuntu2204:~$ sudo cp -v $BUILD_DIR/{ospd-openvas.service,gvmd.service,gsad.service,openvas
d.service} /etc/systemd/system/ && sudo systemctl daemon-reexec && sudo systemctl daemon-reload
'/home/vagrant/build/ospd-openvas.service' -> '/etc/systemd/system/ospd-openvas.service'
'/home/vagrant/build/gvmd.service' -> '/etc/systemd/system/gvmd.service'
'/home/vagrant/build/gsad.service' -> '/etc/systemd/system/gsad.service'
'/home/vagrant/build/openvasd.service' -> '/etc/systemd/system/openvasd.service'
```

Figura 32: Transferindo os arquivos para a as pastas do Systemd.

4.6 Atualizando a base de dados do Greenbone

Passo 11 - Realize o feed-sync através do comando abaixo.

```
sudo /usr/local/bin/greenbone-feed-sync
```

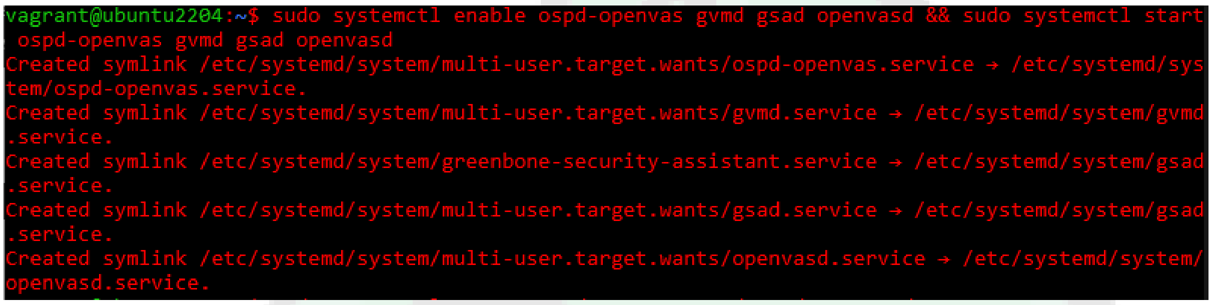


```
⊠ Downloading Notus files from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/vt-data/notus/ to
/var/lib/notus
⊠ Downloading NASL files from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/vt-data/nasl/ to
/var/lib/openvas/plugins
Releasing lock on /var/lib/openvas/feed-update.lock

Trying to acquire lock on /var/lib/gvm/feed-update.lock
Acquired lock on /var/lib/gvm/feed-update.lock
⊠ Downloading SCAP data from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/scap-data/ to
/var/lib/gvm/scap-data
⊠ Downloading CERT-Bund data from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/cert-data/ to
/var/lib/gvm/cert-data
⊠ Downloading gvmd data from rsync://feed.community.greenbone.net/community/data-feed/24.10/ to
/var/lib/gvm/data-objects/gvmd
Releasing lock on /var/lib/gvm/feed-update.lock
```

Figura 33: Executando um Feed-sync.

Passo 12 – Inicie os serviços e configure-os para que sejam ativados automaticamente durante a inicialização do sistema operacional.

```
sudo systemctl enable ospd-openvas gvmd gsad openvasd && sudo systemctl start
ospd-openvas gvmd gsad openvasd
```

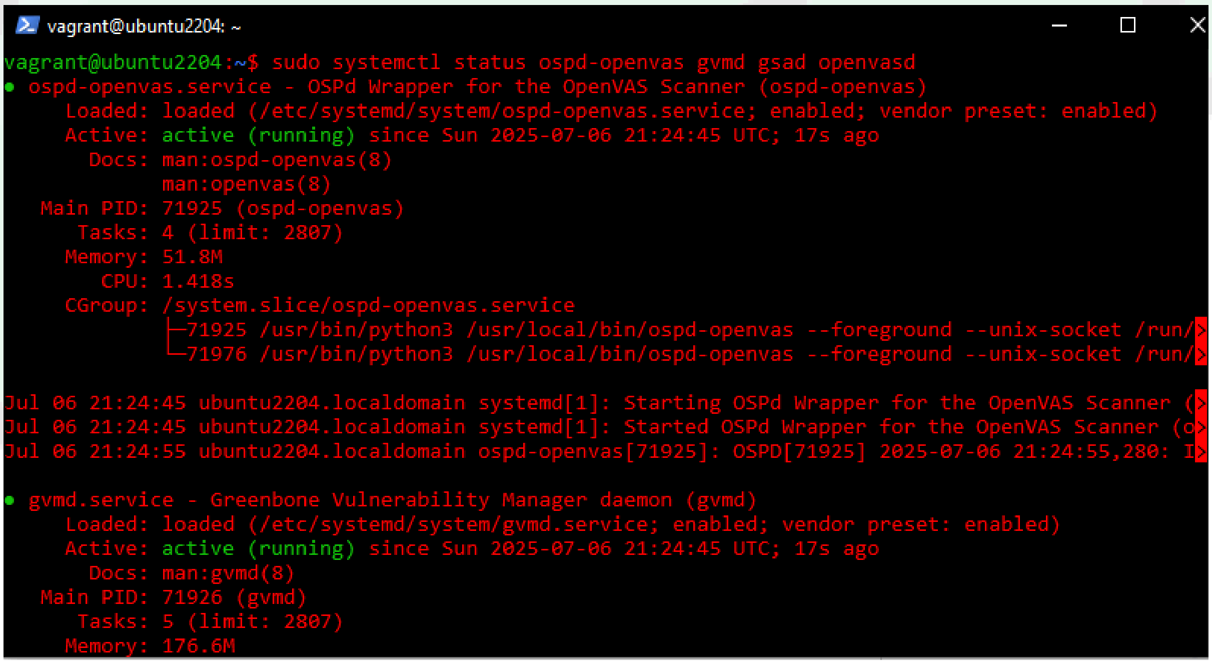


```
vagrant@ubuntu2204:~$ sudo systemctl enable ospd-openvas gvmd gsad openvasd && sudo systemctl start
 ospd-openvas gvmd gsad openvasd
Created symlink /etc/systemd/system/multi-user.target.wants/ospd-openvas.service → /etc/systemd/sys
tem/ospd-openvas.service.
Created symlink /etc/systemd/system/multi-user.target.wants/gvmd.service → /etc/systemd/system/gvmd
.service.
Created symlink /etc/systemd/system/greenbone-security-assistant.service → /etc/systemd/system/gsad
.service.
Created symlink /etc/systemd/system/multi-user.target.wants/gsad.service → /etc/systemd/system/gsad
.service.
Created symlink /etc/systemd/system/multi-user.target.wants/openvasd.service → /etc/systemd/system/
openvasd.service.
```

Figura 34: Executando os serviços e configurando para inicialização automática.

Passo 13 - Confira se está tudo funcional.

```
sudo systemctl status ospd-openvas gvmd gsad openvasd
```



vagrant@ubuntu2204: ~

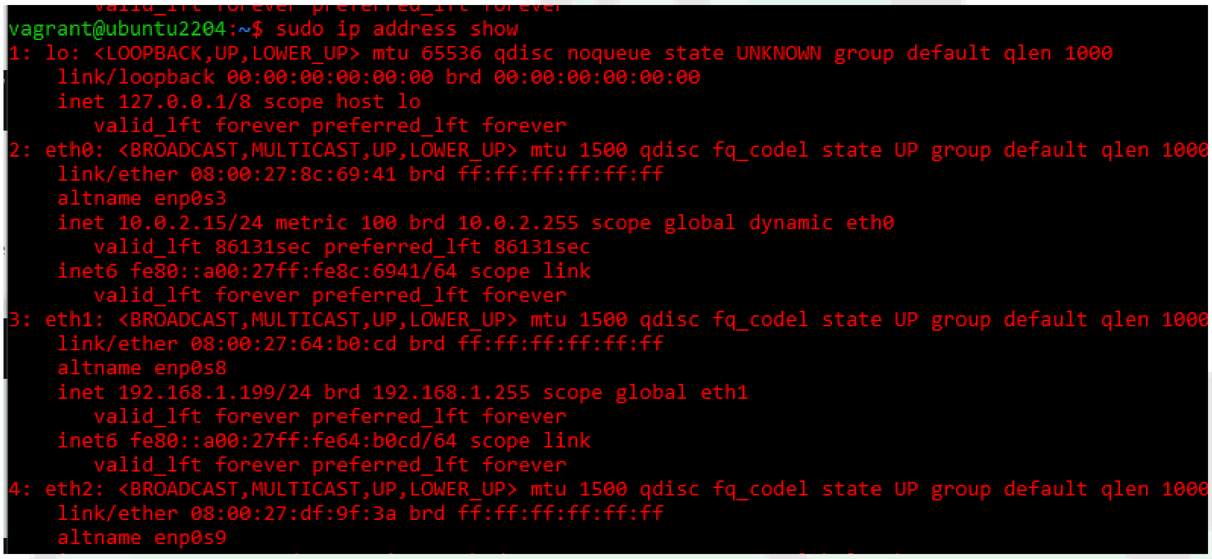
```
vagrant@ubuntu2204:~$ sudo systemctl status ospd-openvas gvmd gsad openvasd
● ospd-openvas.service - OSPd Wrapper for the OpenVAS Scanner (ospd-openvas)
     Loaded: loaded (/etc/systemd/system/ospd-openvas.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2025-07-06 21:24:45 UTC; 17s ago
       Docs: man:ospd-openvas(8)
             man:openvas(8)
   Main PID: 71925 (ospd-openvas)
      Tasks: 4 (limit: 2807)
     Memory: 51.8M
        CPU: 1.418s
     CGroup: /system.slice/ospd-openvas.service
             ├─71925 /usr/bin/python3 /usr/local/bin/ospd-openvas --foreground --unix-socket /run/>
             └─71976 /usr/bin/python3 /usr/local/bin/ospd-openvas --foreground --unix-socket /run/>

Jul 06 21:24:45 ubuntu2204.localdomain systemd[1]: Starting OSPd Wrapper for the OpenVAS Scanner (>
Jul 06 21:24:45 ubuntu2204.localdomain systemd[1]: Started OSPd Wrapper for the OpenVAS Scanner (o>
Jul 06 21:24:55 ubuntu2204.localdomain ospd-openvas[71925]: OSPD[71925] 2025-07-06 21:24:55,280: I>

● gvmd.service - Greenbone Vulnerability Manager daemon (gvmd)
     Loaded: loaded (/etc/systemd/system/gvmd.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2025-07-06 21:24:45 UTC; 17s ago
       Docs: man:gvmd(8)
   Main PID: 71926 (gvmd)
      Tasks: 5 (limit: 2807)
     Memory: 176.6M
```

Figura 35: Serviços do Greenbone em execução.

5. Acessando página Web e verificando a atualização dos feeds.

Passo 1 - Verifique o IP do dispositivo.

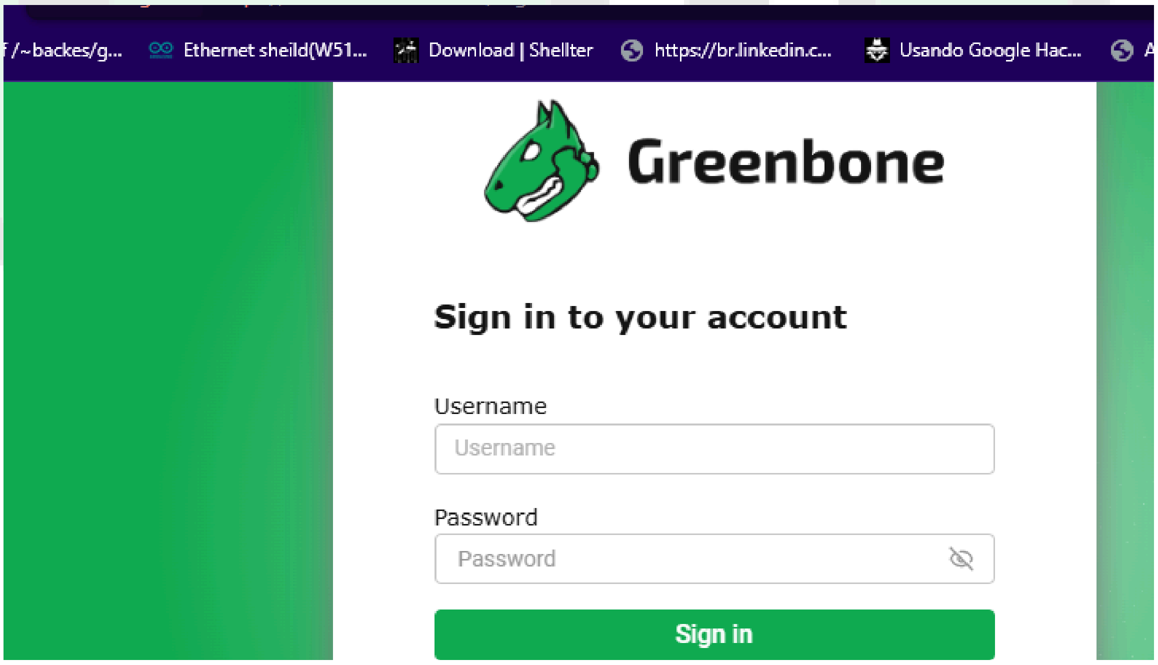
```
sudo ip address show
```



```
      valid_lft forever preferred_lft forever
vagrant@ubuntu2204:~$ sudo ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:8c:69:41 brd ff:ff:ff:ff:ff:ff
    altname enp0s3
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic eth0
       valid_lft 86131sec preferred_lft 86131sec
    inet6 fe80::a00:27ff:fe8c:6941/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:64:b0:cd brd ff:ff:ff:ff:ff:ff
    altname enp0s8
    inet 192.168.1.199/24 brd 192.168.1.255 scope global eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe64:b0cd/64 scope link
       valid_lft forever preferred_lft forever
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:df:9f:3a brd ff:ff:ff:ff:ff:ff
    altname enp0s9
```

Figura 36: Imprimindo os IPs configurados.

Passo 2 - Agora acesse a página web do Greenbone através do IP do dispositivo pela porta 9392, utilize o protocolo https, por exemplo: https://192.168.1.1:9392.



Sign in to your account

Username

Username

Password

Password

Sign in

Figura 37: Página de Login do Greenbone.

Passo 3 - Agora é só realizar o login com o usuário e senha configurados anteriormente para ter acesso às ferramentas.

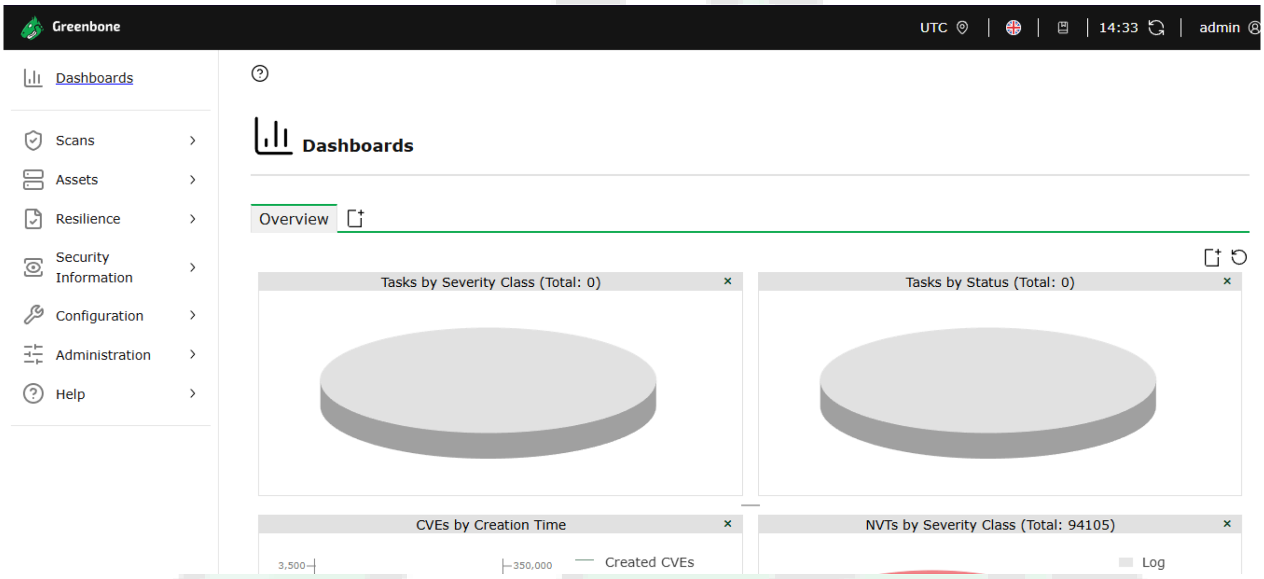


Figura 38: *Dashboard do Greenbone.*

6. Possíveis Problemas que Podem Impedir a Execução do Greenbone

Durante a instalação ou operação do GVM, alguns fatores podem bloquear seu funcionamento. Entre os mais comuns estão:

- **Porta 9392 bloqueada pelo firewall:** Essa porta é usada pelo gsad para oferecer a interface web do Greenbone. Certifique-se de liberá-la nas regras do firewall (ufw, iptables, etc.).
- **Serviços não iniciados corretamente:** Se gvmd, openvas-scanner ou redis-server não estiverem rodando, o GVM pode falhar ao iniciar ou executar scans.
- **Permissões incorretas**: O usuário gvm precisa ter permissões adequadas, incluindo acesso a grupos como redis e permissões de leitura nos diretórios de feed.