Greenbone: installazione e configurazione

Installazione di GCE tramite `docker-compose`

1. Collegamento alla macchina virtuale.
    1. Se si è configurato il client come indicato in 01-greenbone-macchina_virtuale_su_infn-cloud

        ```
        ssh gce
        ```

        altrimenti

        ```
        ssh UTENTE@IP
        ```

        dove al posto di UTENTE ed IP si devono utilizzare rispettivamente il proprio nome utente sulla macchina virtuale e l'IP della macchina virtuale.
2. L'utente deve appartenere al gruppo `sudo`.
3. Aggiornamento dei pacchetti

    ```
    sudo apt update
    sudo apt upgrade -y
    ```

4. **Solo se è stato aggiornato il kernel** eseguire un reboot

    ```
    sudo reboot
    ```

    Attendere il riavvio e collegarsi nuovamente alla macchina

    ```
    ssh gce
    ```

    oppure

    ```
    ssh UTENTE@IP
    ```

    dove al posto di UTENTE ed IP si devono utilizzare rispettivamente il proprio nome utente sulla macchina virtuale e l'IP della macchina virtuale.
5. Intallazione dei pacchetti necessari

    ```
    sudo apt install -y curl
    sudo apt install -y docker.io
    sudo apt install -y python3 python3-pip
    ```

6. Nel caso in cui `~/.local/bin` non sia in `PATH`
   Comando:

   ```
   echo $PATH
   ```

   Output:

   ```
   /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
   /sbin:/bin:/usr/games:/usr/local/games:/snap/bin
   ```

   aggiungere la seguente riga al file `~/.bashrc`

   ```
   PATH=$PATH:~/.local/bin
   ```

   e attivare le modifiche

   ```
   source ~/.bashrc
   ```

   Verificare le modifiche
   Comando:

   ```
   echo $PATH
   ```

   Output:

   ```
   /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
   /sbin:/bin:/usr/games:/usr/local/games:/snap/bin:
   /home/llanzi/.local/bin:
   ```

   In fondo deve comparire : `/home/<NOME UTENTE>/.local/bin`
7. Installazione di `docker-compose` tramite `pip3`

   ```
   python3 -m pip install --upgrade pip
   python3 -m pip install --user docker-compose
   ```

8. Aggiunta del proprio utente al gruppo `docker`

   ```
   sudo usermod -aG docker $USER
   ```

   Nella sessione SSH attiva l'utente non appartiene al gruppo `docker`. È necessario fare un logout e collegarsi nuovamente per avere una sessione in cui l'utente appartenga al gruppo `docker`

   ```
   exit
   ssh llanzi@...
   ```

9. Verifica dell'appartenenza al gruppo `docker` .
   Comando:

```
getent group | grep $USER
```

   Output:

```
llanzi:x:1001:
docker:x:119:llanzi
```

10. Verifica della versione del pacchetto python `requests` tramite pip3
    Comando:

```
pip3 list | grep requests
```

    Output:

```
requests               2.22.0
requests-ntlm          1.1.0
requests-unixsocket    0.2.0
```

    Aggiornamento del pacchetto

```
pip3 install --upgrade requests
```

    Verifica della nuova versione
    Comando:

```
pip3 list | grep requests
```

    Output:

```
requests               2.28.1
requests-ntlm          1.1.0
requests-unixsocket    0.2.0
```

11. Definizione della directory dove salvare il *docker compose file* di GCE

```
export GCE_CONTAINER_DIR=$HOME/greenbone-community-container
```

    e creazione della directory

```
mkdir -p $GCE_CONTAINER_DIR
```

12. Indicare la variabile d'ambiente anche nel file `~/.bashrc` per renderla disponibile nelle sessioni future. Aggiungere la seguente riga al file `~/.bashrc`

```
export GCE_CONTAINER_DIR=$HOME/greenbone-community-container
```

13. Download del *docker compose file* di GCE

```
curl -f -L https://greenbone.github.io/docs/latest/_static/docker-compose-22.4.yml -o $GCE_CONTAINER_DIR/docker-compose.yml
```

14. Download di GCE
Comando:

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-community-edition pull
```

Output:

```
Pulling vulnerability-tests ... done
Pulling notus-data          ... done
Pulling scap-data           ... done
Pulling cert-bund-data      ... done
Pulling dfn-cert-data       ... done
Pulling data-objects        ... done
Pulling report-formats      ... done
Pulling gpg-data            ... done
Pulling redis-server        ... done
Pulling pg-gvm              ... done
Pulling gvmd                ... done
Pulling gsa                 ... done
Pulling ospd-openvas        ... done
Pulling mqtt-broker         ... done
Pulling notus-scanner       ... done
Pulling gvm-tools           ... done
```

15. Esecuzione di GCE
Comando:

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-community-edition up -d
```

Output:

```
Creating network "greenbone-community-edition_default" with the default
driver
Creating volume "greenbone-community-edition_gpg_data_vol" with default
driver
Creating volume "greenbone-community-edition_scap_data_vol" with default
driver
Creating volume "greenbone-community-edition_cert_data_vol" with default
driver
Creating volume "greenbone-community-edition_data_objects_vol" with
default driver
Creating volume "greenbone-community-edition_gvmd_data_vol" with default
driver
Creating volume "greenbone-community-edition_psql_data_vol" with default
driver
Creating volume "greenbone-community-edition_vt_data_vol" with default
driver
Creating volume "greenbone-community-edition_notus_data_vol" with default
driver
Creating volume "greenbone-community-edition_psql_socket_vol" with
default driver
Creating volume "greenbone-community-edition_gvmd_socket_vol" with
default driver
Creating volume "greenbone-community-edition_ospd_openvas_socket_vol"
with default driver
Creating volume "greenbone-community-edition_redis_socket_vol" with
default driver
Creating greenbone-community-edition_redis-server_1        ... done
Creating greenbone-community-edition_mqtt-broker_1         ... done
Creating greenbone-community-edition_gpg-data_1            ... done
Creating greenbone-community-edition_notus-data_1          ... done
Creating greenbone-community-edition_scap-data_1           ... done
Creating greenbone-community-edition_cert-bund-data_1      ... done
Creating greenbone-community-edition_vulnerability-tests_1 ... done
Creating greenbone-community-edition_data-objects_1        ... done
Creating greenbone-community-edition_pg-gvm_1              ... done
Creating greenbone-community-edition_dfn-cert-data_1       ... done
Creating greenbone-community-edition_notus-scanner_1       ... done
Creating greenbone-community-edition_report-formats_1      ... done
Creating greenbone-community-edition_gvmd_1                ... done
Creating greenbone-community-edition_gsa_1                 ... done
Creating greenbone-community-edition_ospd-openvas_1        ... done
Creating greenbone-community-edition_gvm-tools_1           ... done
```

L'elenco dettagliato di tutti i servizi dei container del `docker compose file` è consultabile al link:
https://greenbone.github.io/docs/latest/22.4/container/index.html#description

16. Visualizzazione dei log di GCE (*stream*)

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-community-edition logs -f
```

Per uscire dallao *stream* del log:

```
Ctrl+c
```

17. Modifica della password di amministratore dell'interfaccia web

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-community-edition \
exec -u gvmd gvmd gvmd --user=admin --new-password='1-love-CCR-tooo-much!'
```

18. In questa situazione l'interfaccia web di GCE è esposta tramite http (in chiaro) sulla porta 9392. Per verificarlo eseguire
Comando:

```
netstat -napt
```

Output:

```
[...]
tcp      0      0      0.0.0.0:9392      0.0.0.0:*      LISTEN      -
[...]
```

Se si è installato GCE nella propria postazione di lavoro dotata di un browser web, si può già accedere a GCE collegandosi all'indirizzo: http://127.0.0.1:9392.
Al momento della creazione della macchina virtuale non abbiamo volutamente aperto questa porta all'esterno ma abbiamo aperto la 443 con lo scopo di accedere all'interfaccia web tramite https.

Configurazione dell'interfaccia web di GCE sulla porta 443 (https).

- Alla fine della procedura di installazione indicata nella sezione precedente è possibile accedere all'interfaccia web di GCE in chiaro sulla porta 9392. Se si è installato GCE nella propria postazione di lavoro dotata di un browser web, si può già accedere a GCE collegandosi all'indirizzo: http://127.0.0.1:9392.

- Nel caso invece che l'installazione sia stata fatta su INFN-Cloud o su una macchina virtuale nella propria infrastruttura (ovviamente priva di server X) è necessario esporre GVM in https: **sarebbe veramente deplorevole esporre GVM in chiaro con http (il corso si intitola "Cybersecurity")**.
- Esistono vari modi per spostare GCE su https.
  - Per esempio si può andare a modificare la configurazione del container ma non si segue questa strada perché si preferisce mantenere la configurazione standard data da greenbone in modo da non avere problemi in caso di aggiornamenti futuri.
  - Si preferisce mantenere completamente separata tutta la parte di docker fornita da greenbone e di accedervi tramite un `reverse proxy` con `apache2` (ma si può usare qualsiasi altro server web).

Intallazione di `apache2` e configurazione del `reverse proxy`

1. Intallazione di apache2, abilitazione di ssl, proxy e proxy_http

```
sudo apt install -y apache2
```

2. Abilitazione dei moduli `ssl`, `proxy` e `proxy_http`

```
sudo a2enmod ssl
sudo a2enmod proxy
sudo a2enmod proxy_http
```

3. Disabilitazione della configurazione standard di `apache2`

```
sudo a2dissite 000-default.conf
```

4. Creazione del file di configurazione per impostare il reverse proxy dalla porta 443 alla porta 9392 su localhost.
   Copiare nel file `/etc/apache2/sites-available/gce.conf` quanto segue

```
<IfModule mod_ssl.c>

  <VirtualHost _default_:443>

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    ErrorLog ${APACHE_LOG_DIR}/gce-error.log
    CustomLog ${APACHE_LOG_DIR}/gce-access.log combined

    SSLEngine on
```

```
    SSLCertificateFile  /etc/ssl/certs/ssl-cert-snakeoil.pem
    SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key

    ProxyRequests Off
    ProxyPass / http://127.0.0.1:9392/
    ProxyPassReverse / http://127.0.0.1:9392/

  </VirtualHost>

</IfModule>
```

dove al posto di IP_PUBBLICO deve essere usato l'IP pubblico della macchina virtuale.

5. Abilitare la configurazione

```
a2ensite gce.conf
```

6. Riavviare il server web

```
systemctl restart apache2
```

7. Collegarsi con un growser web all'indirizzo
   `https://IP_PUBBLICO`
   dove al posto di IP_PUBBLICO deve essere usato l'IP pubblico della macchina virtuale. Ovviamente nel browser si deve accettare di utilizzare il certificato autofirmato, ma può essere sostituito semplicemente con un certificato firmato da una certification authority riconosciuta.

Gestione di GCE

Avvio dei container

Comando:

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-
community-edition up -d
```

Output:

```
Starting greenbone-community-edition_vulnerability-tests_1 ... done
Starting greenbone-community-edition_pg-gvm_1               ... done
Starting greenbone-community-edition_mqtt-broker_1          ... done
Starting greenbone-community-edition_cert-bund-data_1       ... done
Starting greenbone-community-edition_gpg-data_1             ... done
Starting greenbone-community-edition_notus-data_1           ... done
```

```
Starting greenbone-community-edition_redis-server_1        ... done
Starting greenbone-community-edition_data-objects_1        ... done
Starting greenbone-community-edition_scap-data_1           ... done
Starting greenbone-community-edition_notus-scanner_1       ... done
Starting greenbone-community-edition_dfn-cert-data_1       ... done
Starting greenbone-community-edition_report-formats_1      ... done
Starting greenbone-community-edition_gvmd_1                ... done
Starting greenbone-community-edition_gsa_1                 ... done
Starting greenbone-community-edition_ospd-openvas_1        ... done
Starting greenbone-community-edition_gvm-tools_1           ... done
```

Spegnimento dei container

Comando:

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-
community-edition stop
```

Output:

```
Stopping greenbone-community-edition_gsa_1          ... done
Stopping greenbone-community-edition_gvmd_1         ... done
Stopping greenbone-community-edition_mqtt-broker_1  ... done
Stopping greenbone-community-edition_redis-server_1 ... done
Stopping greenbone-community-edition_pg-gvm_1       ... done
```

Aggiornamento dei feed (Greenbone Community Feed)

```
# download dei container con i feed
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-
community-edition pull notus-data vulnerability-tests scap-data dfn-cert-data
cert-bund-data report-formats data-objects

# esecuzione dei container per caricare i dati nei volumi
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-
community-edition up -d notus-data vulnerability-tests scap-data dfn-cert-data
cert-bund-data report-formats data-objects
```

Verifica delle operazioni di aggiornamento

- ospd-openvas VTs
    - L'inizio dell'aggiornamento viene indicato con la seguente voce nei log

```
Loading VTs. Scans will be [requested|queued] until VTs are loaded.
This may take a few minutes, please wait...
```

- La fine dell'aggiornamento viene indicato con la seguente voce nei log

```
Finished loading VTs. The VT cache has been updated from version X
to Y.```
```

- Per verificare le operazioni di aggiornamento si può usare il seguente comando

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
greenbone-community-edition logs | grep -E 'Loading VTs. Scans will
be|Finished loading VTs'
```

- gvmd VTs
  - L'inizio dell'aggiornamento viene indicato con la seguente voce nei log

```
OSP service has different VT status (version X) from database
(version (Y), Z VTs). Starting update ...
```

  - La fine dell'aggiornamento viene indicato con la seguente voce nei log

```
Updating VTs in database ... done (X VTs).
```

  - Per verificare le operazioni di aggiornamento si può usare il seguente comando

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
greenbone-community-edition logs | grep -E 'OSP service has
different VT status|Updating VTs in database'
```

- gvmd SCAP data
  - L'inizio dell'aggiornamento viene indicato con la seguente voce nei log

```
update_scap: Updating data from feed
```

  - La fine dell'aggiornamento viene indicato con la seguente voce nei log

```
update_scap_end: Updating SCAP info succeeded
```

  - Per verificare le operazioni di aggiornamento si può usare il seguente comando

```
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
greenbone-community-edition logs | grep -E 'update_scap: Updating
```

```
data from feed|update_scap_end: Updating SCAP info succeeded'
```

- gvmd CERT data
  - L'inizio dell'aggiornamento viene indicato con la seguente voce nei log

    ```
    sync_cert: Updating data from feed
    ```

  - La fine dell'aggiornamento viene indicato con la seguente voce nei log

    ```
    sync_cert: Updating CERT info succeeded.
    ```

  - Per verificare le operazioni di aggiornamento si può usare il seguente comando

    ```
    docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
    greenbone-community-edition logs | grep -E 'sync_cert: Updating data
    from feed|sync_cert: Updating CERT info succeeded'
    ```

- GVMD Data
  - Port list

    ```
    docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
    greenbone-community-edition logs | grep 'Port list'
    ```

  - Report format

    ```
    docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
    greenbone-community-edition logs | grep 'Report format'
    ```

  - Scan config

    ```
    docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p
    greenbone-community-edition logs | grep 'Scan config'
    ```

Altre operazioni sui container che potrebbero essere utili

Lista dei container in esecuzione

```
docker ps
```

Lista di tutti i container

```
docker ps -a
```

Accesso ad un container

```
CONTAINER_NAME=...
docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-
community-edition exec $CONTAINER_NAME /bin/bash
```

Per esempio, usando uno dei seguenti valori per `$CONTAINER_NAME`

```
CONTAINER_NAME="ospd-openvas"
CONTAINER_NAME="gsa"
CONTAINER_NAME="gvmd"
CONTAINER_NAME="notus-scanner"
CONTAINER_NAME="mqtt-broker"
CONTAINER_NAME="redis-server"
CONTAINER_NAME="pg-gvm"

docker-compose -f $GCE_CONTAINER_DIR/docker-compose.yml -p greenbone-
community-edition exec $CONTAINER_NAME /bin/bash
```