Fl0wer Evaluation OVA image

The Fl0wer Evaluation OVA image (shortly, the “eval”) is a quick way to deploy the Fl0wer platform solution in your virtualized environment to quickly evaluate it.

The image has been tested on the following virtualization platforms (it’s created and converted from a Proxmox KVM):

- VMWARE ESXi 7.0.3

The image is based on a stable Debian 12 with the following software already installed and preconfigured :

- Fl0wer 3.5 runtime and development environment with Python 3.12 (evaluation version)
- Clickhouse OLAP database server 25.12.2.54
- Apache 2.4.66-1~deb12u1

The scope of the “eval” image is to test Fl0wer to see if it can suit your environment and *could* not be the best deployment solution. Its sole scope is this: evaluate; if you want to put it in your production environment, unless you have a support contract and a software license, you are on your own.

Evaluation version limitations

The evaluation version has the following limitations:

1.Multi-threading is disabled. You will have only 1 thread processing incoming data.
2.The Flow Matrix is limited to 50 entries (no limits in the full version)
3.Does not handle Netflow V9 and IPFIX (no limits in the full version)
4.Does not handle IPv6 Flows (no limits in the full version)
5.Allows only Dynamic SSLv23 In-Memory Certificates for the API (regenerated after each daemon restart). The full version uses static TLS 1.2 certificate for the API.

Apart from this, it has no time limits, no limits on the number of flow exporters or anything else.

For more information about Fl0wer, refer to the website https://fl0wer.me and the related user manual.

Hardware requirements

The Evaluation runs ok with 4Gb of RAM memory (my lab runs on an 8Gb RAM VM in Proxmox). As a rule of thumb: the more resources you give it, the better the performance.
You should tune-up Clickhouse to better use all the memory you will provide it with.
Regarding the storage, as a personal suggestion, in 2026, don't use nothing worse than an SSD. NVME storage is much way better. Worse will work, but will be slow as death.
Also regarding the CPU, my homelab runs on an ODROID-H3 which has an Intel(R) Pentium(R) Silver N6005 @ 2.00GHz, I am sure you have better hardware than me :-)

System services listening:
Port 80/tcp: Fl0wer Web Interface (you go with your browser here)
Port 8000/tcp: Raw Streamlit web interface (using web sockets) – you can use this too.
Port 7443/tcp: Fl0wer API
Port 8123/tcp: ClickHouse Server API
Port 22/tcp: SSH for remote server management
Port 2055/udp: Fl0wer Netflow/IPfix Listener (traffic here is redirected by samplicate)
Port 6344/udp: Fl0wer sFlow V5 Listener (traffic here is redirected by samplicate)
Port 2056/udp: Samplicator Netflow/IPfix Listener (forwards to Fl0wer, you can add other targets if you want)
Port 6343/udp: Samplicator sFlow V5 Listener (forwards to Fl0wer, you can add other targets if you want)

To add other targets, just check /usr/local/bin/samplicate.sh and eventually systemctl restart samplicate.

System users:
root – password: fl0werr0x
fl0wer – password fl0werr0x
You can change the passwords using the standard Linux command passwd. For remote access, login via SSH as fl0wer user and use the su – command. Hey, it's a Debian 12, you can mess around as you want! Remove the other unused users if needed but not the system ones like fl0wer, clickhouse, apache, etc.

Application user:
Fl0wer user fl0wer – password: fl0werr0x (you can manager users with the */opt/fl0wer/bin/flcli-*user** commands. Remember to adjust any script that contains the password.
Clickhouse server user default password: fl0werr0x – The password hash is stored in */etc/clickhouse-server/users.d/default-password.xml* – Follow the Clickhouse procedures to change it if you need so.

System services to check:
The OVA, to work, needs some services that are defined in the directory */etc/systemd/system* and are:

- samplicate.service: The Samplicate process for forwarding the flow traffic to Fl0wer or other systems of your choiche if needed
- dnsmonster.service: The DNS Monster service (https://dnsmonster.dev/) that injects the DNS requests into the Clickhouse database. If you have a spare passive interface that can be used for passive traffic analysis, DNSmonster can "sniff" all the DNS traffic (queries and responses) and pump it in the Clickhouse OLAP database.
- fl0wer.service (-> */opt/fl0wer/systemd/fl0wer.service*): The Fl0wer product – receives and processes all Netflow/Ipfix/sFlow traffic to be feeded in Clickhouse by the fl0wpumper
- fl0wpumper.service (-> */opt/fl0wer/systemd/fl0wpumper.service*): Takes in input the CSV files created by Fl0wer for injection in Clickhouse, ELK, etc. Full source code available for easy customization.
- fl0wer-webui.service (-> */opt/fl0wer/systemd/fl0wer-webui.service*): The Fl0wer dashboard. Full source code available for easy customization.
- In */lib/systemd/system/clickhouse-server.service* you will find the clickhouse server process, which is where all the data is going to be stored.

All services are managed by *systemd*.

Cleanup
For a quick & easy cleanup of the system, in the */root* folder there are the following scripts:

1. *createdb.sh*
2. *create_dnsmonster.sh*
3. *create_rsyslog.sh*

You can run them in order to cleanup the Clickhouse database and recreate Database and tables.
Example:
root@fl0wer-eval:~# ./createdb.sh && ./create_dnsmonster.sh && ./create_rsyslog.sh

Note: if you change the password for the Clickhouse or the Fl0wer user, remember to fix the scripts too.

Fl0wer customization
Refer to the user manual, anyway all files are under */opt/fl0wer/etc and /opt/fl0wer/iplist.*
As a minimum rule of thumb, at least customize the *fl0wer_internal_networks.conf* providing it with the network subnets you have in your site.

Storage space
The OVA image was sized as a 30Gb disk, which should be sufficient (in most cases) to showcase the Fl0wer Platform. Anyway if you have an army of Flow Exporters and tons of data incoming, I suggest you to add a virtual disk (better in form of NVME storage) and using standard linux LVM commands to enlarge the filesystem. Fl0wer is self-contained in */opt/fl0wer* and the *data* directory grows, specially if you enable realtime LUA scripts, maybe you want/can move it onto its separate space, you are free to do what you prefer. Consider that where the storage also really grows is in the */var/lib/clickhouse*, where the DB is stored, so having a single filesystem could not be a so bad idea at all. Do your math.

Network Configuration
The network configuration is in classic Debian */etc/network/interfaces* system file. Change it to suit your needs.
It first tries to DHCP, then if not available reverts to the following configuration:

```
# The primary network interface
allow-hotplug eth0
auto eth0
iface eth0 inet dhcp
    dhcp-timeout 180
    post-up /etc/network/dhcp-fallback.sh

# Fallback static configuration
iface eth0 inet static
        address 192.168.1.30/24
        gateway 192.168.1.20
        dns-nameservers 1.1.1.1
        down ip route del 10.0.0.0/8 via 192.168.1.10 dev eth0
        up ip route add 10.0.0.0/8 via 192.168.1.10 dev eth0
```

It’s Debian, you can adjust it for your needs. Use a fixed IPv4 address and point all your flow exporters to it on the 2056 and 6343 UDP ports.

Centralized Syslog
The rsyslog installed in this distribution can receive log messages on standard port 514 and store it in the Clickhouse database server, just check in */etc/rsyslog.d/50-clickhouse.conf* and uncomment all the lines. It is disabled by default to avoid filling up your storage, but you can use it and it’s integrated in the Fl0wer dashboard.

Packet filtering
A local packet filter is enabled and installed by using firewall-cmd (which in turns uses nftables).

This is the default configuration:

```
root@fl0wer-eval:~# firewall-cmd --list-all
public
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
  services: dhcpv6-client ssh
  ports: 2056/udp 2055/udp 6343/udp 6344/udp 80/tcp 443/tcp 7443/tcp 9000/tcp 514/tcp 514/udp
  protocols:
  forward: yes
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Reports
There's a cronjob for the fl0wer user that runs every hour to produce handy HTML and PDF reports under */var/www*, choose your way to publish them or to use them if you want.

Hacking
Except for the core of Fl0wer daemon, everything is provided with sources. You can model the Fl0wer Platform as best as you want. The dashboard is in */opt/fl0wer/web and it's written in Python 3.12 and Streamlit.* Create the views you want and enjoy, and if you use it seriously, buy the license !

Hope you will enjoy Fl0wer !
gilberto