FRAFOS ABC SBC Installation Guide

Release 5.5.2

FRAFOS GmbH

Jun 04, 2025

Table of Contents

The ABC SBC is distributed in form of a container of OCI type. It can be run on operating system of customer choice, if the OS supports running of that container type.

FRAFOS also offers an Amazon cloud based solution where the ABC SBC is running as an instance in AWS (EC2). This is by far the fastest installation, the software can be started by several clicks. See amazon.

FRAFOS can also provide a hardware based solution with preinstalled ABC SBC software on a reference hardware, see *Hardware Requirements* for more details.

Chapter 1

Hardware Requirements

FRAFOS ABC SBC is provided as container, internally based on Debian 12 64bit operating system (x86_64 architecture).

Capacity and performance of the system depends mainly on the number and type of processors (CPU), available operating memory (RAM) and the number and performance of network cards (NIC).

There are no specific constraints for vendors of hardware and components, but we do have some suggestions and recommendations for the used hardware and its settings. Generally amount of memory and CPU power increases system resilience against load peaks, Ethernet cards with high packet rate facilitate high media anchoring throughput and fast solid-state drives facilitate WAV and PCAP recording.

Minimum hardware:

- CPU: 1x processor - 64bit architecture
- RAM: 4 GB
- NIC: 1x 1Gb network card
- HDD: 10 GB

Recommended / reference hardware: Fujitsu Primergy RX1330 (M3 or M4)

- CPU: Intel® Xeon® processor E3-1200 family, 4GHz
- RAM: 64 GB (DDR4 2666 MHz, dual channel)
- NIC: 2 x Intel 1Gb adapter
- SSD: 2 x 256 GB

For more details about system capacity and dimensioning, see Sec. Sec-Dimensioning.

Chapter 2

Deployment Modes

According to the system dimensioning and high-availability requirements, ABC SBC can be deployed in different modes:

2.1 Single Node Mode

In single node mode a single ABC SBC container is used. It is necessary to connect this node to a CCM module which provides GUI for administration of system and serves as a configuration master to all connected ABC SBC nodes. The CCM module is distributed as a single container and can be either deployed on the same host together with ABC SBC node or on a different server.

2.2 High Available (HA) Pair Mode

IMPORTANT note: up to ABC SBC release 4.1, it was using HA solution based on Pacemaker. The ABC SBC 4.2 was a transitional release that removed Pacemaker based HA solution. The new HA solution based on keepalived was introduced in ABC SBC 4.3 release.

HA pair ABC SBC installation is formed by two physically identical servers running in an active/hot-standby configuration. Only the active (HA master) server processes signaling and media traffic. In case of any device or network failure, the internal management system performs a failover where the originally standby (HA backup) machine becomes active. Switching the active and standby operation modes is also useful during the upgrades of the system.

Both the HA master and backup servers share virtual IP addresses (VIPs) and communicate with each other over the "Internal Management Interface" - IMI. The HA backup server can check the network availability of the HA master server. Once the HA backup server determines that the HA master server is no longer reachable over IMI interface then the backup server will assume the role of HA master and take over the VIPs used for receiving and sending the media and signaling messages.

Further, the HA master server replicates state information about running sessions to the HA backup machine. Thereby, after a failover the backup server will be able to continue processing already established calls and the failure of the server will not result in dropping of already established calls. Note however that calls are replicated after they are established and calls unanswered yet may be dropped. Also only signaling over connection-less transport protocols is certain to reach the Call Agents as transport protocol context gets lost during failover.

Note: status about non running SEMS process reported by SNMP from nodes in 'BACKUP' state should be ignored

2.3 Cluster based solution

For very high traffic and performance requirements, ABC SBC instances (in a single or HA pair mode) can create a cluster. A SIP load balancer is put in front of these cluster nodes so as to distribute the SIP traffic to a particular ABC SBC instances.

Chapter 3

Container Installation

Before actual installation admin should consider enabling coredumps on the host, see coredumpsref. It may be beneficial to have them enabled in advance, to ease later troubleshooting but it needs to be considered that whole host and all containers running there will be influenced.

3.1 Podman

In this section it is described the OCI container installation process under podman for the ABC SBC and the Cluster Config Manager. The procedure is based on podman 4.3.1 used on Debian 12. In case podman version 5.x is used, this handbook provides different network configuration as podman 5.x contains `--route` option.

Please refer to the official documentation for more detailed information.

3.1.1 Installing podman

Since Debian 12 the podman package is available in official repositories and can be installed on host with Debian OS via:

```
% apt install podman
```

3.1.2 OCI images download

Start by downloading the OCI images directly from Frafos's docker registry (registry.frafos.net) using the following:

```
% podman pull registry.frafos.net/abc/sbc:5.5
% podman pull registry.frafos.net/abc/ccm:5.5
```

Frafos's OCI tagging strategy is the following:

- *5.0.[0,100]*: the tag matches the image exact version (*5.0.1*, *5.0.2*)
- *5.0*: alias to the latest *5.0.X* image for the major release
- *5.0-XX*: alias to an exact *5.0.X* image (*5.0-rc1*, *5.0-dev*)

One may also run the following to pull the desired exact images:

```
% # exact minor release
% podman pull registry.frafos.net/abc/sbc:5.0.42
% # test the release candidate
% podman pull registry.frafos.net/abc/ccm:5.0-rc1
```

(continues on next page)

(continued from previous page)

```
% # test the "latest" 5.0
% podman pull registry.frafos.net/abc/ccm:5.0
```

The access to container registry requires authentication using username and password. The account can be self-created at https://registry.frafos.net/, but the account has to be then assigned to proper project by Frafos. Please contact Frafos support, if your account is not approved yet for access to the ABC SBC repository project.

To log in to the Frafos repository, use the following command. You will be prompted to enter the username and password associated with your customer account:

```
% podman login registry.frafos.net
```

3.1.3 Persistent data

It is highly recommended to use persistent `/data` directory for the container, where all configuration, that needs to be preserved across upgrades, is stored.

This can be reached either by mounting an external directory from the host into the container's `/data` directory:

```
% mkdir -p /var/data/ccm
```

or by using a *podman volume* for Cluster Config Manager and ABC SBC separately:

For Cluster Config Manager installation:

```
% podman volume create ccm-data
```

For ABC SBC installation:

```
% podman volume create sbc-data
```

In the first case, the storage is fully under administrator's control and can be accessed not only from containers, but directly by other processes as well. This can be beneficial for downloading CDRs from ABC SBC, for backups generated from host or for mounting a specific partition.

In the second case, the storage is fully managed by podman and can be accessed only from containers or via podman commands. This may be sometimes limiting and thus in general rather not the recommended way.

3.1.4 Podman Container Installation

There are many ways how to configure networking with podman which may vary use case by use case. For simplification we will focus on the most common scenario only: separate MACVLAN networks for management and VoIP traffic. This configuration allows multiple containers (Cluster Config Manager and ABC SBC) running on the same host or on different hosts, depending on desired performance and other requirements.

In networking matter, if the platform is AWS, Azure or GCP, it is not needed to have network configuration as these platforms are not supporting MACVLAN. Only the "host" network type is supported.

Podman installation for AWS - Azure - GCP

After pulling Cluster Config Manager image to the host, the container can be created by the following command:

```
% podman create --name ccm --hostname ccm --tz=local --tty \
 --mount type=volume,src=ccm-data,target=/data,rw=true \
 --cap-add=AUDIT_CONTROL --cap-add=AUDIT_WRITE --cap-add=NET_RAW \
 --network host \
 registry.frafos.net/abc/ccm:5.5
```

For ABC SBC container creation:

```
% podman create --name sbc --hostname sbc --tz=local --tty \
 --mount type=volume,src=sbc-data,target=/data,rw=true \
 --cap-add=NET_RAW \
 --pids-limit 65536 \
 --network host \
 registry.frafos.net/abc/sbc:5.5
```

Notes about parameters:

- The `--pids-limit` can be updated according to the threads number that is used by sems. By default it is 2048, however with higher number of threads (used by sems) sbc requires higher limit.
- The `--tz=local` enables container to use the same timezone with host.
- When HA is used, adding `--cap-add=SYS_NICE` to the SBC container could prevent false failovers that could happen when the system is under heavy load.

After container creation, a service should be created by the following commands to manage container, and then this will create a service which needs to be started and enabled:

```
% cd /etc/systemd/system/
```

For Cluster Config Manager:

```
% podman generate systemd --new -f -n ccm
% systemctl start container-ccm
% systemctl enable container-ccm
```

For ABC SBC:

```
% podman generate systemd --new -f -n sbc
% systemctl start container-sbc
% systemctl enable container-sbc
```

Please note that each change (container name, adding a network, adding capabilities) must be added in service file (/etc/systemd/system/container-ccm.service).

To check logs, this command can be run:

```
% podman logs -f ccm
% podman logs -f sbc
```

Regular podman installation

Interface configuration like IP assignment, DNS, etc. can be passed to the container by podman.

Deciding interface numbers should be according to the topology. In basic topology Cluster Config Manager should have only a management (mgmt) interface. On the other hand, ABC SBC should have these interfaces by default:

mgmt: will be used for communication between Cluster Config Manager, Monitor and other ABC SBC in case HA is used. For mgmt interface, one IP should be assigned on the host, one IP should be assigned for container.

internal: will be used for SIP/MEDIA communication with internal sip servers. For internal interface, there is no need to assign an IP on the host, only for container.

external: will be used for SIP/MEDIA communication with public sip servers (like providers). For external interface, there is no need to assign an IP on the host, only for container.

If more interfaces are needed for cluster, then external2, external3 etc. can be added as well. Obviously, the names of the interfaces can be updated as customers wish.

For Cluster Config Manager a specific routing is not needed as it supposed to have only one interface. If additional routing rule is necessary, please check the note related to routing part under ABC SBC network configuration. For Cluster Config Manager container creation, only one network is required for the Cluster Config Manager, even if the Manager and Monitor can be installed together on the same host. A bridged network is used here with the driver netavark. The definition is made using a simple command:

```
% podman network create mgmt
```

Display a list of existing Podman networks:

```
% podman network ls
```

An individual network can also be displayed for checking purposes. The exact definition within the container definition is displayed here:

```
% podman network inspect mgmt
```

Then please set proper parameters while creating the Cluster Config Manager container:

```
% podman create --name ccm --hostname ccm --tz=local --tty \
  --mount type=volume,src=ccm-data,target=/data,rw=true \
  --cap-add=NET_RAW --cap-add=AUDIT_CONTROL --cap-add=AUDIT_WRITE \
  --network mgmt:interface_name=eth0 \
  -p 443-444:443-444 \
  --dns=172.22.31.2 \
  registry.frafos.net/abc/ccm:5.5
```

Notes about parameters:

- The `--pids-limit` can be updated according to the threads number that is used by sems. By default it is 2048, however with higher number of threads (used by sems) sbc requires higher limit.
- The `--tz=local` enables container to use the same timezone with host.
- The `NET_RAW` is needed for raw sockets usage and additionally for troubleshooting purposes (for example to allow `ping` utility).
- The `--dns=none` flag tells podman not to create `/etc/resolv.conf` in the container.

Please note, that to make per-interface DNS configuration working in an ABC SBC container, it is necessary to allow resolvconf when performing the SBC initial config. To make it working in Cluster Config Manager, a CCM configuration option “Enable to use resolvconf for dns” needs to be set.

For ABC SBC container network creation:

Several networks are required for the SBC, here no bridged network is used, but macvlan network. As driver netavark is used. The Podman networks are defined via the command line as examples for podman 4.3.1 and podman 5.4.1, please note that customer should decide for each subnet according to the customer network.

For podman 4.3.1, `--internal` must be added to avoid adding default gateway for the network.

Management network mgmt 10.10.10.0/24 via enp0s8

```
% podman network create --driver macvlan --opt parent=enp0s8 --gateway 10.10.10.1 --
↪internal --subnet 10.10.10.0/24 mgmt
```

Voice network internal 10.10.20.0/24 via enp0s9:

```
% podman network create --driver macvlan --opt parent=enp0s9 --gateway 10.10.20.253 --
↪internal --subnet 10.10.20.0/24 internal
```

Voice network (with default gateway) External 10.10.30.0/24 via enp0s10:

```
% podman network create --driver macvlan --opt parent=enp0s10 --gateway 10.10.30.253 -
↪-subnet 10.10.30.0/24 external
```

For podman 5.4.1, `-o no_default_route=1` must be added to avoid adding default gateway for the network.

Management network mgmt 10.10.10.0/24 via enp0s8

```
% podman network create --driver macvlan -o no_default_route=1 --opt parent=enp0s8 --
↪gateway 10.10.10.1 --subnet 10.10.10.0/24 mgmt
```

Voice network internal 10.10.20.0/24 via enp0s9:

```
% podman network create --driver macvlan -o no_default_route=1 --opt parent=enp0s9 --
↪gateway 10.10.20.253 --subnet 10.10.20.0/24 internal
```

Voice network (with default gateway) External 10.10.30.0/24 via enp0s9:

```
% podman network create --driver macvlan --opt parent=enp0s10 --gateway 10.10.30.253 -
↪-subnet 10.10.30.0/24 external
```

A note related to routing:

If additional routing rules are required; For podman 4.3.1, a script can be added under /data/init/pre inside the container. This will help to run anything in pre-init part of container initiation. A sample script might be like this:

routing.sh:

```
#!/bin/bash

# new subnet route to internal network
ip route add 172.16.5.0/24 via 10.10.20.1 dev eth1
ip route add 172.16.6.0/24 via 10.10.20.1 dev eth1

# new subnet route to mgmt network
ip route add 192.168.20.0/24 via 10.10.10.1 dev eth0
```

For podman 5.4.1, `--route` option can be used during network creation: A static route in the format <destination in CIDR notation>,<gateway>,<route metric (optional)>. This route will be added to every container in this network. It can be specified multiple times if more than one static route is desired.

A sample:

```
% podman network create --driver macvlan --opt parent=enp0s9 --gateway 10.10.30.253 --
↪subnet 10.10.30.0/24 --route 10.40.20.0/24,10.10.30.1 external
```

Then please set proper parameters while creating the container:

```
% podman create --name sbc --hostname sbc --tz=local --tty \
  --mount type=volume,src=sbc-data,target=/data,rw=/true \
  --cap-add=NET_ADMIN --cap-add=NET_RAW \
  --pids-limit 65536 \
  --network mgmt:interface_name=eth0,ip=10.10.10.100 \
  --network internal:interface_name=eth1,ip=10.10.20.101 \
  --network external:interface_name=eth2,ip=10.10.30.102 \
  --dns=172.22.31.2 \
  registry.frafos.net/abc/sbc:5.5
```

Notes about parameters:

- The `--pids-limit` can be updated according to the threads number that is used by sems. By default it is 2048, however with higher number of threads (used by sems) sbc requires higher limit.
- The `--tz=local` enables container to use the same timezone with host.
- The `NET_RAW` is needed for raw sockets usage and additionally for troubleshooting purposes (for example to allow `ping` utility).
- The `--dns=none` flag tells podman not to create `/etc/resolv.conf` in the container.

After container creation, a service should be created by the following commands to manage container, and then this will create a service which needs to be started and enabled:

```
% cd /etc/systemd/system/
```

For Cluster Config Manager:

```
% podman generate systemd --new -f -n ccm
% systemctl start container-ccm
% systemctl enable container-ccm
```

For ABC SBC:

```
% podman generate systemd --new -f -n sbc
% systemctl start container-sbc
% systemctl enable container-sbc
```

Please note that each change (container name, container image, adding capabilities) must be updated in service file (/etc/systemd/system/container-ccm.service).

To check logs, this command can be run:

```
% podman logs -f ccm
% podman logs -f sbc
```

3.1.5 Container management

To list running containers use:

```
% podman ps
```

To list even stopped containers:

```
% podman ps -a
```

To list volumes:

```
% podman volume ls
```

To list all images:

```
% podman image ls
```

To start, stop or restart container:

```
% systemctl start container-ccm
% systemctl stop container-ccm
% systemctl restart container-ccm
```

Executing commands within the container

To open a shell inside the container, use the following command on the host server:

```
% podman exec -it sbc bash
```

Another commands can be executed directly similar way:

```
% podman exec -it sbc ip route
```

Checking journal

As ABC SBC does not contain journal anymore, container's systemd journal or podman logs can be checked:

```
% podman logs -f ccm
```

or:

```
% journalctl -u container-ccm
```

3.1.6 Upgrade Procedure

For a container upgrade, please start by pulling the newest images:

```
% podman pull registry.frafos.net/abc/sbc:5.5
% podman pull registry.frafos.net/abc/ccm:5.5
```

You may check the latest images metadata using:

```
% podman image ls
REPOSITORY                        TAG     IMAGE ID      CREATED       SIZE
registry.frafos.net/abc/ccm       5.5     bc3c1b61316a  2 hours ago   1.03 GB
registry.frafos.net/abc/sbc       5.5     574b44e60f30  5 hours ago   1.25 GB
<none>                            <none>  655cf4d30cbf  24 hours ago  1.25 GB
<none>                            <none>  8e0eb0df41c0  24 hours ago  1.03 GB
```

To re-create and re-start the container (ABC SBC in this case), using the newest image, update the version part (end of the "ExecStart" line) in container service:

```
% vi /etc/systemd/system/container-sbc.service
```

and then reload the daemon and restart the container service:

```
% systemctl daemon-reload
% systemctl restart container-sbc
```

One may then remove the unused images (old, untagged variant), using:

```
% podman image prune
WARNING! This will remove all dangling images.
Are you sure you want to continue? [y/N] y
8e0eb0df41c0c9c8cb6c8154b5fc7f4979869ede92febc7f220b4b6a08ebf133
655cf4d30cbf018198f096bf059283eda27c9f9b0073f92ae748af74316e6be4
```

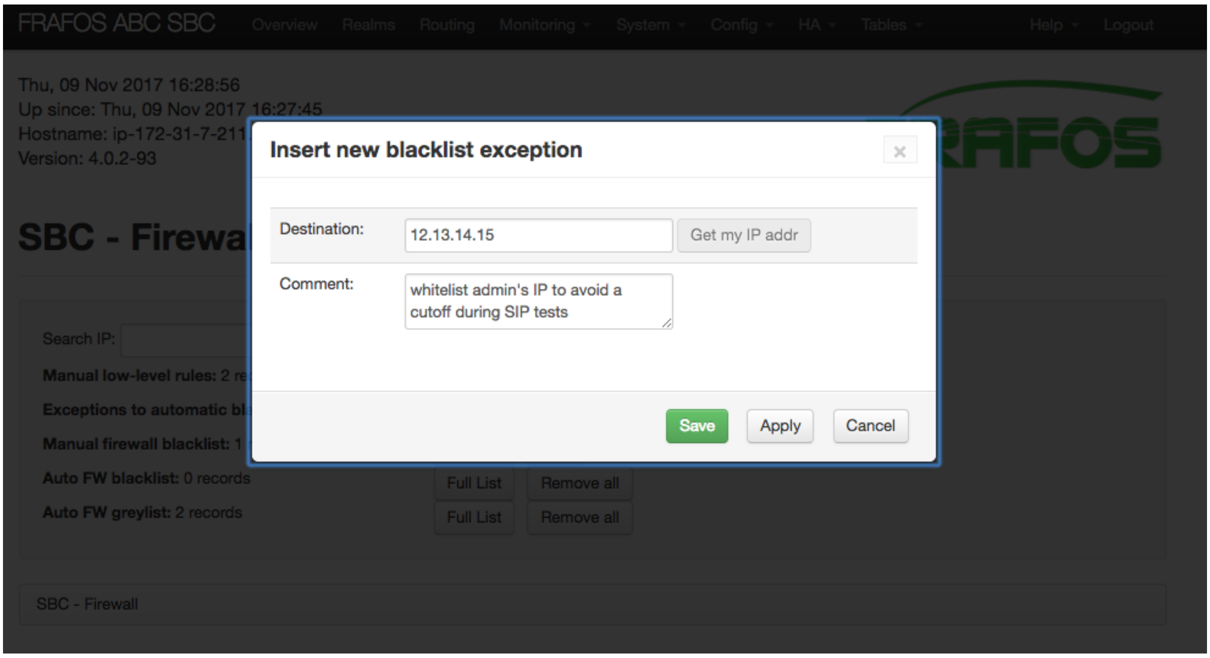
Chapter 4

Post-installation steps

Note: IMPORTANT: AFTER THE INSTALLATION PROCESS IS COMPLETE AND BEFORE CONFIGURATION AND TESTING BEGINS WE URGE YOU TO WHITELIST THE IP ADDRESS FROM WHICH THE ABC SBC WILL BE ADMINISTERED.

Failure to whitelist the administrator's IP address may – especially during the initial configuration and testing – easily block the administrative access to the machine. Various automated blacklisting techniques can block the whole IP address if they spot unexpected traffic from the IP address. See more details in Section Sec-Abuse.

To whitelist the IP address, visit the administrative GUI under **"Config → Firewall → Exceptions to automatic Blacklists → Add"** as shown in the Figure bellow:



Thu, 09 Nov 2017 16:28:56
Up since: Thu, 09 Nov 2017 16:27:45
Hostname: ip-172-31-7-211
Version: 4.0.2-93

Insert new blacklist exception

Destination: 12.13.14.15

Comment: whitelist admin's IP to avoid a cutoff during SIP tests

Fig. 1: Warning: Whitelist Administrator's IP Address