Evidian

SafeKit User's Guide

High Availability Software for Critical Applications

Overview

Subject	This document covers all the phases of the SafeKit implementation: architecture, installation, tests, administration & troubleshooting, support, and command line interface.	
Intended Readers	***Architectures***	"High availability architectures" page 15 "SafeKit cluster in the cloud" page 315
	Installation	"Installation" page 25
	Console	"The SafeKit web console" page 35 "Securing the SafeKit web service" page 177
	Advanced configuration	"Cluster.xml for a SafeKit cluster configuration" page 227 "Userconfig.xml for a module configuration" page 235 "User application scripts for the module configuration" page 293 "Examples of userconfig.xml and user scripts" page 299
	Administration	"Mirror module administration" page 95 "Farm module administration" page 105 "Command line interface" page 141 "Advanced administration" page 153
	Support	"Tests" page 69 "Troubleshooting" page 109 "Access to Evidian support" page 133 "Log Messages Index" page 337
	Other	"Table of Contents" page 5 "Third-Party Software" page 333
Release	SafeKit 7.5	
Supported OS	Windows and Linux; for a detailed list of supported OS, see here	

Web Site	Evidian marketing site: Evidian support site:	http://www.evidian.com/safekit https://support.evidian.com/safekit
Ref	39 A2 19MC 01	

If you have any comments or questions related to this documentation, please mail us at **institute@evidian.com**

Evidian disclaims the implied warranties of merchantability and fitness for a particular purpose and makes no express warranties except as may be stated in its written agreement with and for its customer. In no event is Evidian liable to anyone for any indirect, special, or consequential damages.

The information and specifications in this document are subject to change without notice. Consult your Evidian Marketing Representative for product or service availability.

Table of Contents

1. High availability architectures

1.1 SafeKit cluster definition

A SafeKit cluster is a set of servers where SafeKit is installed and running.

All servers belonging to a given SafeKit cluster share the same cluster configuration (list of servers and networks used) and communicate with each other's to have a global view of SafeKit modules configurations. The same server can not belong to many SafeKit clusters.

Setting the cluster configuration is a prerequisite to SafeKit modules installation and configuration since the 7.2 release of SafeKit and of the web console. The cluster configuration is set through the web console as described in section 3.2 page 37. The web console provides the ability to administer one or more SafeKit clusters.

1.2 SafeKit module definition - application integration

A SafeKit module is associated with an application. A module is customizable by the user, and it defines the behavior of the high availability solution for the application. Different modules can be defined for different applications.

In practice, an application module is an easy-to-setup file that contains:

- ➩ a main configuration file `userconfig.xml`, which lists networks used for communication between servers, files to replicate in real time (for a mirror module), virtual IP configuration, network load balancing criteria (for a farm module) and more...
- ➩ application stop and start scripts

SafeKit offers two types of modules detailed in this chapter:

- ➩ the mirror module
- ➩ the farm module

Combining multiple application modules allows the implementation of advanced architectures:

- ➩ active/active: 2 mirror modules backuping each other
- ➩ N-1: N mirror modules with a single backup
- ➩ mixed farm and mirror: mixing network load balancing, file replication and failover

1.3 Mirror module: synchronous real time file replication and failover

1.3.1 File replication and failover

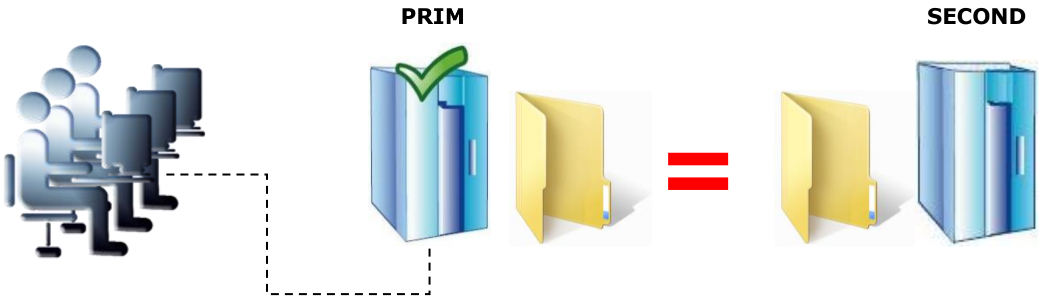
The mirror architecture is a primary-backup high-availability solution that is suitable for all applications. The application runs on a primary server and is restarted automatically on a secondary server if the primary server fails.

The mirror architecture can be configured with or without file replication. With its file-replication function, this architecture is particularly suitable for providing high availability for back-end applications with critical data to protect against failure. Indeed, the secondary server data are highly synchronized with the primary server and the failover is done on the secondary server from the most up-to-date data. If the application availability is more critical than the application data synchronization, the default policy can be relaxed by allowing a failover on the secondary server when the time elapsed since the last synchronization is below a configurable delay.

Microsoft SQL Server.Safe, MySQL.Safe, and Oracle.Safe are examples of "mirror" type application modules. You can write your own mirror module for your application, based on the generic module Mirror.Safe.

The failover mechanism works as follows.

1.3.2 Step 1. Normal operation



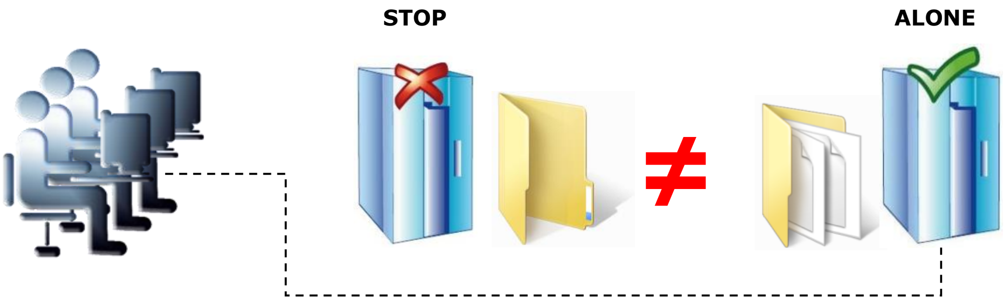
For replication, only the names of file directories are configured in SafeKit. There are no pre-requisites on the disk organization for the two servers. Directories to replicate can be located in the system disk.

Server 1 (`PRIM`) runs the application.

SafeKit replicates files opened by the application. Only the changes made by the application in the files are replicated in real time across the network, thus limiting traffic.

Thanks to the synchronous replication of file write operations on the disks of both servers, no data is lost in case of failure.

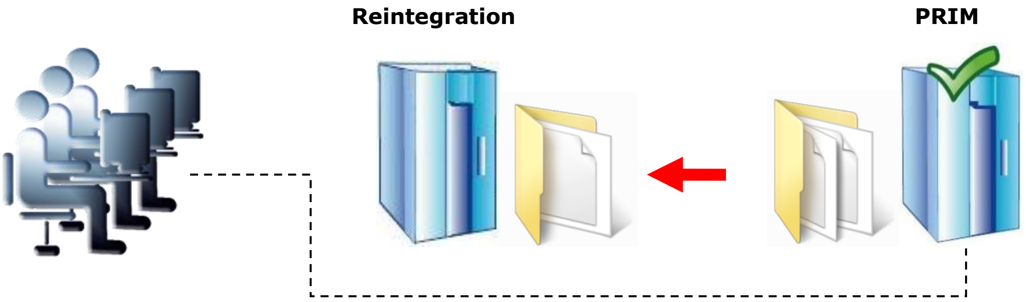
1.3.3 Step 2. Failover



When Server 1 fails, Server 2 takes over. SafeKit switches the cluster's virtual IP address and restarts the application automatically on Server 2. The application finds the files replicated by SafeKit in the identical state they were when Server 1 failed, thanks to the synchronous replication. The application continues to run on Server 2, locally modifying its files, which are no longer replicated to Server 1.

The switch-over time is equal to the fault-detection time (set to 30 seconds by default) plus the application start-up time. Unlike disk replication solutions, there is no delay for remounting file systems and running recovery procedures.

1.3.4 Step 3. Failback and reintegration



Failback involves restarting Server 1 after fixing the problem that caused it to fail. SafeKit automatically resynchronizes the files, updating only the files that were modified on Server 2 while Server 1 was stopped.

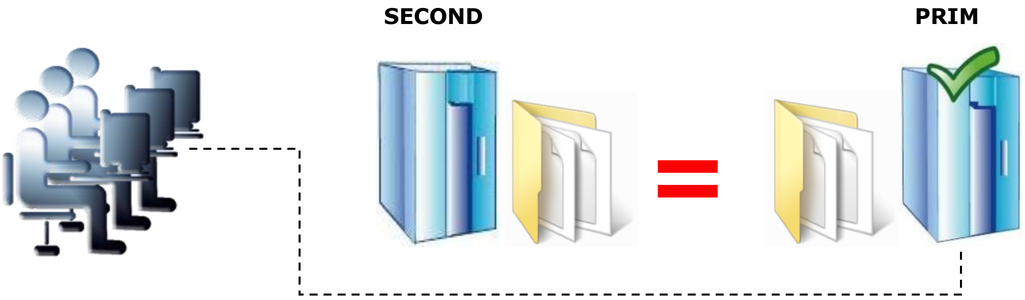
This reintegration takes place without disturbing the applications, which can continue to run on Server 2. This is a major feature that differentiates SafeKit from other solutions, which require you to stop the applications on Server 2 to resynchronize Server 1.

To optimize file reintegration, different cases are considered:

1. The module must have completed the reintegration (on the first start of the module, it runs a full reintegration) before enabling the tracking of modification into bitmaps
2. If the module was cleanly stopped on the server, then at restart of the secondary, only the modified zones of modified files are reintegrated, according to a set of modification tracking bitmaps.
3. If the secondary crashed (power off) or was incorrectly stopped (exception in nfsbox replication process), the modification bitmaps are not reliable, and are therefore discarded. All the files bearing a modification timestamp more recent than the last known synchronization point minus a graceful delay (typically one hour) are reintegrated.

4. A call to the special command `second fullsync` triggers a full reintegration of all replicated directories on the secondary when it is restarted.
5. If files have been modified on the primary or secondary server while SafeKit was stopped, the replicated directories are fully reintegrated on the secondary

1.3.5 Step 4. Return to normal operation



After reintegration, the files are once again in mirror mode, as in step 1. The system is back in high-availability mode, with the application running on Server 2 and SafeKit replicating file updates to the backup Server 1.

If the administrator wants to run the application on Server 1, he/she can execute a `swap` command either manually at an appropriate time, or automatically through configuration.

1.3.6 Synchronous, fault-tolerant replication that loses no data when a server fails

There is a significant difference between synchronous replication, as offered by the SafeKit mirror solution, and asynchronous replication traditionally offered by other file replication solutions.

With synchronous replication, when a disk IO is performed by the application or by the file cache system on the primary server onto a replicated file, SafeKit waits for the IO acknowledgement from the local disk and from the secondary server, before sending the IO acknowledgement to the application or to the file system cache.

The synchronous, in real time, replication of files updated by an application eliminates the loss of data in case of server failure. Synchronous replication ensures that any data committed on a disk by a transactional application is also present on the secondary server.

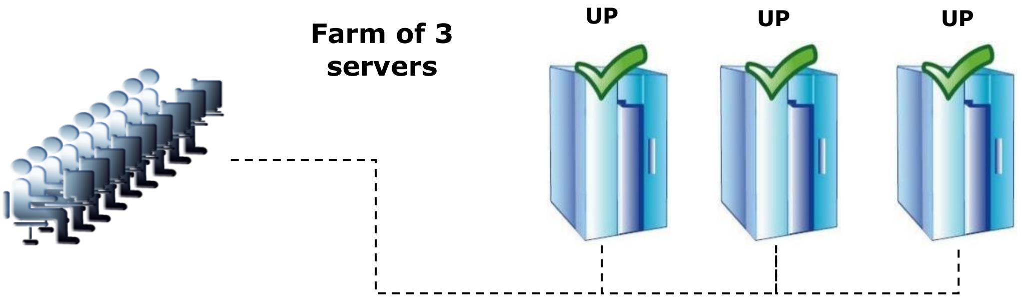
The bandwidth required to implement synchronous data replication is in the order of magnitude of a typical modern LAN, or extended LAN between two computer rooms located a few kilometers apart.

With asynchronous replication implemented by other solutions, the IOs are placed in a queue on the primary server but the primary server does not wait for the IO acknowledgments of the secondary server. So, the data that did not have time to be copied across the network on the second server is lost if the first server fails. In particular, a transactional application loses committed data in case of failure. Asynchronous replication can be used for data replication through a low-speed WAN, to back up data remotely over more than 100 kilometers.

SafeKit provides an asynchronous solution with no data loss, ensuring the asynchrony not on the primary machine but on the secondary one. In this solution, SafeKit always waits for the acknowledgement of the two machines before sending the acknowledgement to the application or the system cache. But on the secondary, there are 2 options asynchronous or synchronous. In the asynchronous case (option <rfs async="second">), the secondary sends the acknowledgement to the primary upon receipt of the IO and writes to disk after. In the synchronous case (<rfs async="none">), the secondary writes the IO to disk and then sends the acknowledgement to the primary. The async="none" mode is required if we consider a simultaneous double power outage of two servers, with inability to restart the former primary server and requirement to re-start on the secondary.

1.4 Farm module: network load balancing and failover

1.4.1 Network load balancing and failover



The farm architecture provides both network load balancing, through transparent distribution of network traffic, and software and hardware failover. This architecture provides a simple solution for increasing system load. The same application runs on each server, and the load is balanced by the distribution of network activity between the different servers of the farm.

Farm architecture accommodates/implements well with front-end applications like web services. Apache_farm.Safe and Microsoft IIS_farm.safe are examples of farm application modules. You can make your own farm module, modified to suit your application, from the generic module Farm.safe.

1.4.2 Principle of a virtual IP address with network load balancing

The virtual IP address is configured locally on each server of the farm. The input traffic for this address is split among them at low level by a filter inside each server's kernel.

The load balancing algorithm inside the filter is based on the identity of the client packets (client IP address, client TCP port). Depending on the identity of the client packet input, a single filter instance in a server farm transmits the packet to the upper network layers; the other filter instances in other servers drop it. Once a packet is accepted by the filter on a server, only the CPU and memory of this server are used by the application that responds to the request of the client. The output messages are sent directly from the application server to the client.

If a server fails, the SafeKit membership protocol reconfigures the filters in the farm to re-balance the traffic on the remaining available servers.

1.4.3 Load balancing for stateful or stateless web services

With a stateful server, there is session affinity. The same client must be connected to the same server on multiple HTTP/TCP sessions to retrieve its context from the server. In this case, the SafeKit load balancing rule is configured on the client IP address. Thus, the same client is always connected to the same server on multiple TCP sessions. And different clients are distributed across different servers in the farm. This configuration is used when there is a need for session affinity.

With a stateless server, there is no session affinity. The same client can be connected to different servers in the farm on multiple HTTP/TCP sessions; because there is no context stored locally on a server from one session to another. In this case, the SafeKit load balancing rule criteria is the TCP client session identity. This configuration is the best solution to distribute sessions between servers, but it can only loadbalance a TCP service without session affinity.

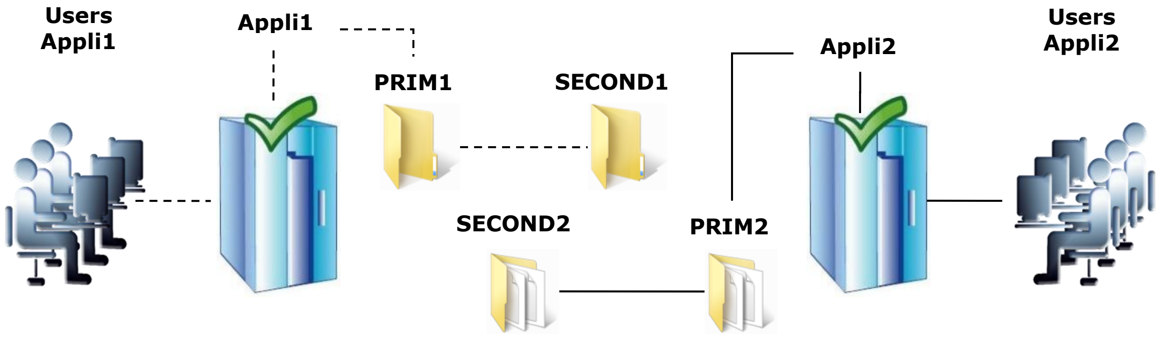
Other load balancing algorithms are available for UDP services.

1.5 Combining mirror and farm modules

1.5.1 Active/Active: 2 mirror modules backuping each other

Two active servers mirroring each other

In an active / active architecture, there are two servers and two mirror application modules in mutual takeover (Appli1.Safe and Appli2.Safe). Each application server is a backup of the other server.



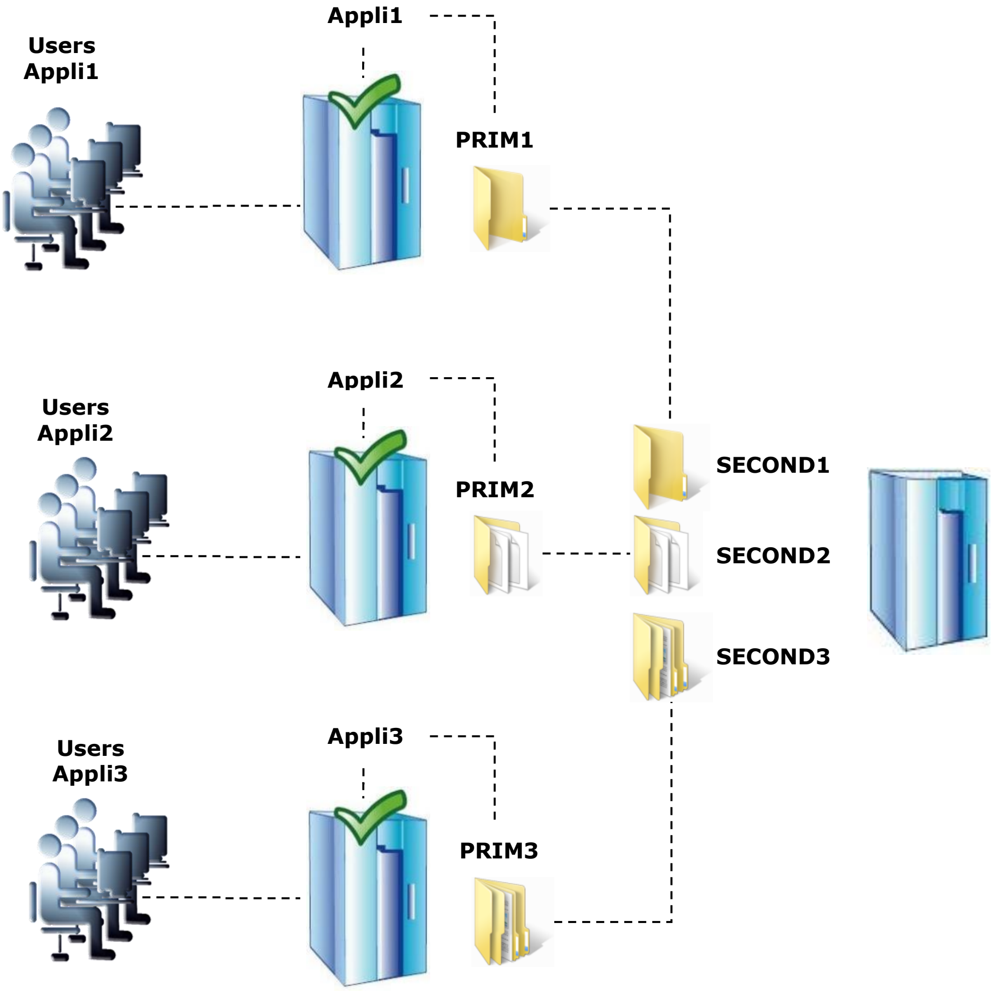
If one application server fails, both applications will be active on the same physical server. After restart of the failed server, its application will run again on its default primary server.

A mutual takeover cluster is a more economical solution than two separate mirror clusters, because there is no need to invest in back-up servers that will spend most of their time sitting idle waiting for the primary server to fail. Note that during a failure, the remaining server must be able to handle the combined workload of both applications.

1.5.2 N-to-1: N mirror modules with a single backup

Shared backup for multiple active servers

In N-to-1 architecture, there are N mirror application modules installed on N primary servers and one backup server.



If one of the N active servers fails, the single backup server restarts the module of the failed server. Once the problem is fixed and the failed server is restarted, the application switches back to its original server.

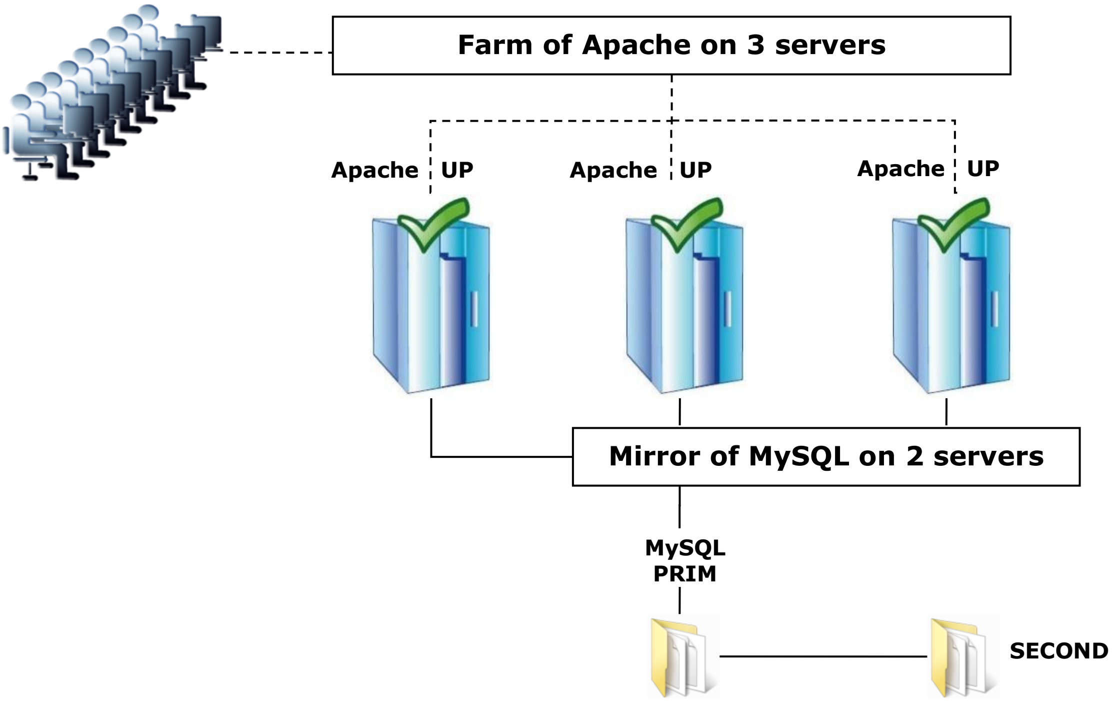
In case of failure, unlike the active/active architecture, the backup server doesn't have to handle a double workload when a primary server fails. Assuming that there is only one failure at a time - the solution can support multiple primary server failures at the same time, but in this case the single back-up server will have to handle the combined workload of all the failed servers.

1.5.3 Mixed farm/mirror: network load balancing, file replication, failover

Network load balancing, file replication and failover

You can mix farm and mirror application modules on the same cluster of servers.

This option allows you to implement a multi-tier application architecture, such as Apache_farm.Safe (farm architecture with load balancing and failover) and MySQL.safe (mirror architecture with file replication and failover) on the same application servers.



As a result, load balancing, file replication and failover are managed coherently on the same servers. Specific to SafeKit, this mixed architecture is unique on the market!

1.6 The simplest high availability cluster in the cloud

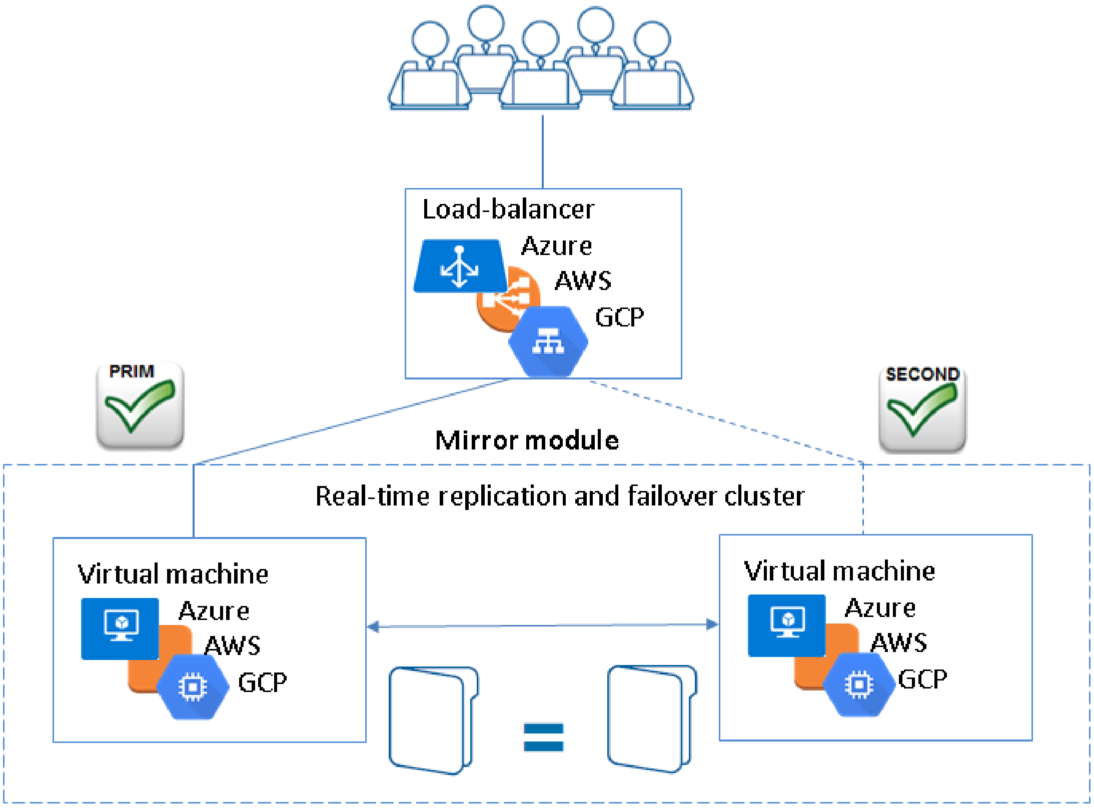
SafeKit brings in the Microsoft Azure, Amazon AWS and Google clouds the simplest solution for a high availability cluster. It can be implemented on existing virtual machines or on a new virtual infrastructure, that you create by simply clicking on a button that deploys and configures everything for you in Azure or AWS clouds.

For a full description, see section 16 page 315.

1.6.1 Mirror cluster in Microsoft Azure, Amazon AWS and Google GCP

SafeKit brings in the Azure, Aws and GCP clouds the simplest solution for a high availability cluster with real-time replication and failover (mirror module).

For a quick start, refer to mirror cluster in Azure, mirror cluster in AWS or mirror cluster in GCP.



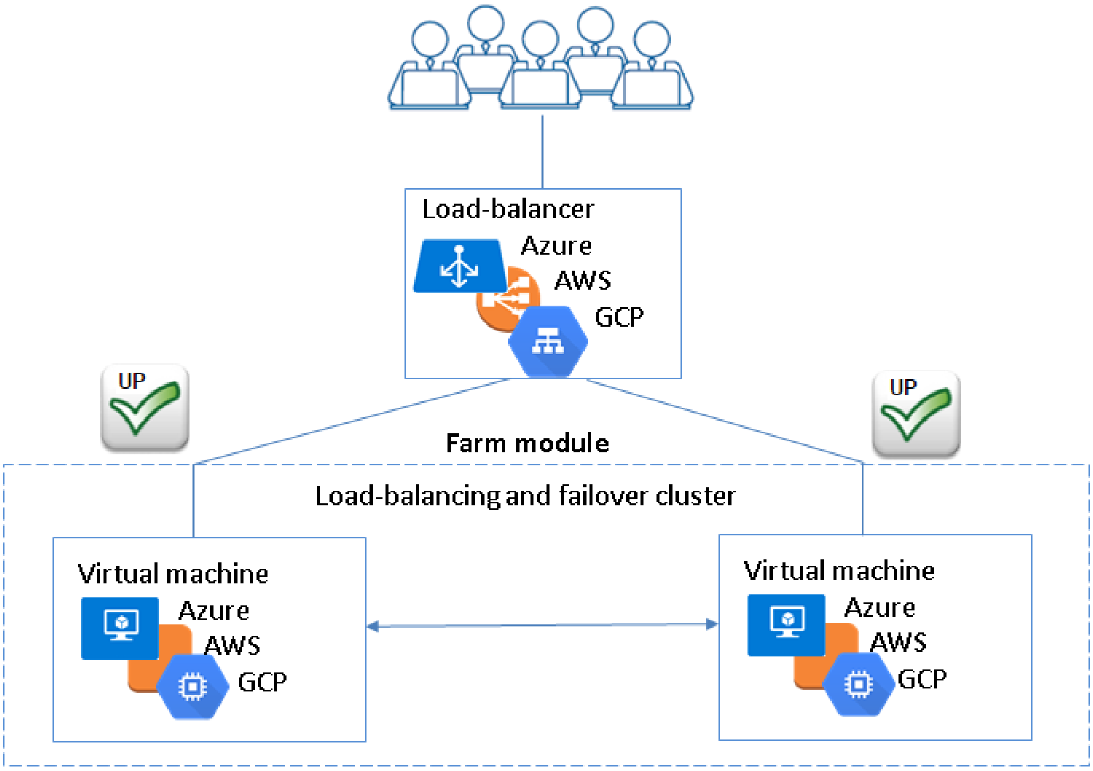
- the critical application is running on the `PRIM` server
- users are connected to a primary/secondary virtual IP address which is configured in the cloud load balancer
- SafeKit brings a generic checker for the load balancer. On the `PRIM` server, the checker returns OK to the load balancer and NOK on the `SECOND` server
- in each server, SafeKit monitors the critical application with process checkers and custom checkers
- SafeKit automatically restarts the critical application when there is a software failure or a hardware failure thanks to restart scripts

- ➪ SafeKit makes synchronous real-time replication of files containing critical data
- ➪ a connector for the SafeKit web console is installed in each server. Thus, the high availability cluster can be managed in a very simple way to avoid human errors

1.6.2 Farm cluster in Microsoft Azure, Amazon AWS and Google GCP

SafeKit brings in the Azure, AWS and Google clouds the simplest solution for a high availability cluster with load balancing and failover (farm module).

For a quick start, refer to farm cluster in Azure, farm cluster in AWS or farm cluster in GCP.



- ➪ the critical application is running in all servers of the farm
- ➪ users are connected to a virtual IP address which is configured in the cloud load balancer
- ➪ SafeKit brings a generic checker for the load balancer. When the farm module is stopped in a server, the checker returns NOK to the load balancer which stops the load balancing of requests to the server. The same behavior happens when there is a hardware failure
- ➪ in each server, SafeKit monitors the critical application with process checkers and custom checkers
- ➪ SafeKit automatically restarts the critical application in a server when there is a software failure thanks to restart scripts
- ➪ a connector for the SafeKit web console is installed in each server. Thus, the load balancing cluster can be managed in a very simple way to avoid human errors

2.Installation

2.1 SafeKit install

2.1.1 Download the package

1. Connect to https://support.evidian.com/safekit
2. Go to `<Version 7.5>/Platforms/<Your platform>/Current versions`
3. Download the 64-bit package

2.1.2 Installation directories and disk space provisioning

SafeKit is installed in:

`SAFE`	➩ in Windows
	`SAFE=C:\safekit`
	if `%SYSTEMDRIVE%`=C:
	➩ in Linux
	`SAFE=/opt/safekit`
		Minimum free disk space: 80MB

`SAFEVAR`	➩ in Windows
	`SAFEVAR= C:\safekit\var`
	if `%SYSTEMDRIVE%`=C:
	➩ in Linux
	`SAFEVAR=/var/safekit`
		Minimum free disk space: 20MB + at least 20MB (up to 3 GB) per module for dumps

2.1.3 Install procedure

2.1.3.1 On Windows as administrator

- SafeKit package install
  1. log-in as administrator
  2. locate the downloaded file `safekitwindows_7_5_x_y.msi`
  3. install in interactive mode by double-clicking it and go through the installer wizard

  or

3. install in non-interactive mode, by executing in PowerShell console as administrator:

   ```
   msiexec /qn /i safekitwindows_7_5_x_y.msi
   ```

➩ Firewall setup

1. in a PowerShell console as administrator
2. run `SAFE/private/bin/firewallcfg add`

   It configures the Microsoft firewall for SafeKit. For details or other firewalls, see section 10.3 page 156

➩ Web service initialization for the SafeKit console and distributed commands

This step is mandatory to initialize the default configuration of the web service. Since SafeKit 7.5, the web service requires authentication to access the service. This script makes it easy to implement by initializing it with the `admin` user and the given password `pwd`, for example.

1. in a PowerShell console as administrator
2. run `SAFE/private/bin/webservercfg -passwd pwd`

This then allows to access to all the web console's features, by logging in with `admin/pwd`, and to run distributed commands. For details, see 11.2.1 page 179.

Important The password must be identical on all nodes that belong to the same SafeKit cluster. Otherwise, web console and distributed commands will fail with authentication errors.

Note On upgrade, this step can be skipped if it has already been done in a previous version of SafeKit 7.5. If it is reapplied, it will reset the password with the new value.

2.1.3.2 On Linux as root

➩ SafeKit package install

1. log-in as root
2. locate the downloaded file `safekitlinux_7_5_x_y.bin`

   auto extractible zip file
3. run `chmod +x safekitlinux_7_5_x_y.bin`
4. run `./safekitlinux_7_5_x_y.bin`

   it extracts the package and the `safekitinstall` script
5. install in interactive mode by executing ./`safekitinstall`

   During the installation:

   ✓ reply to "Do you accept that SafeKit automatically configure the local firewall to open these ports (yes|no)?"

If you answer `yes`, it configures `firewalld` or `iptable` Linux firewall for SafeKit. For details or other firewalls, see section 10.3 page 156.

✓ reply to "Please enter a password or "no" if you want to set it later :"

This step is mandatory to initialize the default configuration of the web service. Since SafeKit 7.5, the web service requires authentication to access the service.

It initializes it with the `admin` user and the given password `pwd`, for instance. It then allows to access to all the web console's features, by logging in with `admin/pwd`, and run distributed commands. For details, see 11.2.1 page 179.

Important The password must be identical on all nodes that belong to the same SafeKit cluster. Otherwise, web console and distributed commands will fail with authentication errors.

or

5. install in non-interactive mode, by executing:

```
safekitinstall -q
```

Use the option `-nofirewall` for disabling the firewall automatic setup

Use the option `-passwd pwd` for initializing the web service authentication (where `pwd` is the password set for the `admin` user)

⇨ Firewall setup

No action required when firewall automatic configuration has been performed during install. Otherwise see section 10.3 page 156.

⇨ Web service initialization for the SafeKit console and distributed commands

No action required when the web service initialization has been performed during install. Otherwise, see section 11.2.1 page 179.

2.1.4 Use the SafeKit console or command line interface

Once installed, the SafeKit cluster must be defined. Then modules can be installed, configured, and administered. All these actions can be done with the SafeKit console or the command line interface.

⇨ The SafeKit console

1. start a web browser
2. connect it to the URL `http://servername:9010` (where `servername` is the name or IP address of one of the SafeKit nodes)
3. in the login page, enter `admin` as user's name and the password you gave on initialization as password (e.g., `pwd`). Then click on Connect
4. the loaded page contains all the tabs that correspond to the Admin role by default

   Admin role: Configuration, Control, Monitoring and Advanced Configuration

For details see section 3 page 35

- ➩ The SafeKit command line interface

  It relies on the single command `safekit` located into the `SAFE` directory (in Windows, `SAFE=C:\safekit` if `%SYSTEMDRIVE%`=C: ; in Linux, `SAFE=/opt/safekit`). Almost all `safekit` commands can be applied locally or on a list of nodes in the SafeKit cluster. This is called distributed command.

  For details, see section 9 page 141.

2.1.5 SafeKit license keys

- ➩ If you do not install any license keys, the product will stop every 3 days
- ➩ You can download a one-month trial key (which is accepted on any hostname/any OS) from the following address: http://www.evidian.com/safekit/requestevalkey.php
- ➩ To obtain permanent keys see section 8.2 page 134
- ➩ Save the key into the `SAFE/conf/license.txt` file on each server
- ➩ Check the key conformance with the command `safekit level`

2.1.6 System specific procedures and characteristics

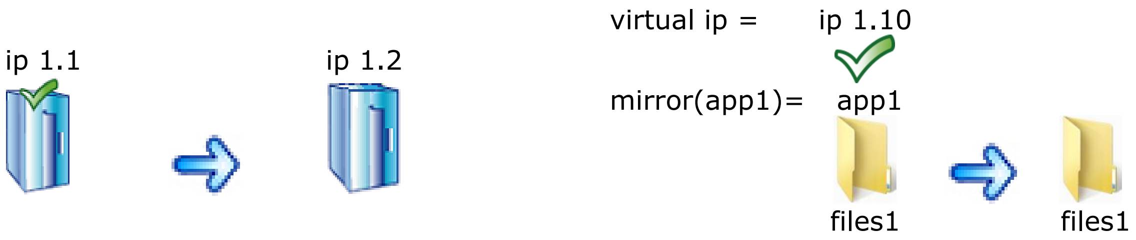
2.1.6.1 Windows

- ➩ Apply a special procedure to properly stop SafeKit modules at machine shutdown and to start `safeadmin` service at boot: see section 10.4 page 161.
- ➩ For network interfaces with teaming and with SafeKit load balancing, it is necessary to uncheck "Vip" on physical network interfaces of teaming and keep it checked only on teaming virtual interface.

2.1.6.2 Linux

- ➩ In RedHat/CentOS the following packages are required:
  - ✓ for SafeKit core: coreutils, sed, gawk, bind-utils
  - ✓ for file replication: nfs-utils
  - ✓ for load balancing: make, gcc, kernel-devel and elfutils-libelf-devel in CentOS 8
- ➩ The user "safekit" and a group "safekit" are created: all users of the "safekit" group + "root" can execute SafeKit command lines
- ➩ For a farm with SafeKit load balancing on a bonding interface, no ARP should be set in the bonding configuration. Otherwise the association <virtual IP address, invisible virtual MAC address> is broken in client ARP caches with physical MAC address of the bonding interface: see section 4.3.4 page 80
- ➩ For a mirror, if using file replication, remove the logwatch package (rpm -e logwatch); otherwise NFS service and SafeKit are stopped every night

2.2 Mirror installation recommendation



2.2.1 Hardware prerequisites

- 2 servers with the same Operating System
- Supported OS: https://support.evidian.com/supported_versions/#safekit
- Disk drive with write-back cache recommended for the performance of the IOs

2.2.2 Network prerequisites

- 1 physical IP address per server (ip 1.1 and ip 1.2)
- If you need to set a virtual IP address (ip 1.10), both servers must be in the same IP network with the standard SafeKit configuration (LAN or extended LAN between two remote computer rooms). For setting a virtual IP address with servers in different IP networks, see section 13.5.3 page 243.

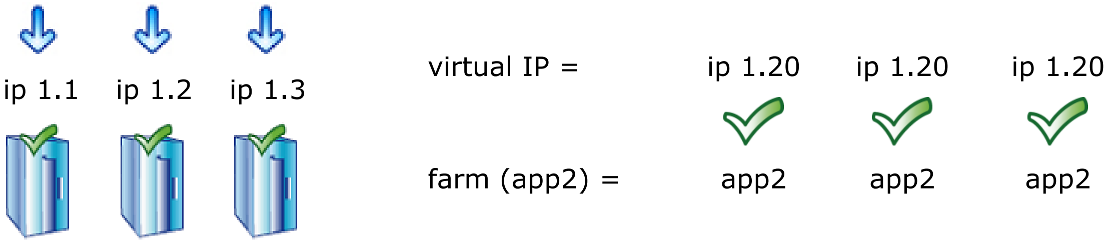
2.2.3 Application prerequisites

- The application is installed and starts on both servers
- Application can be started and stopped using command lines
- On Linux, command lines like `service "service" start|stop` or `su -user "appli-cmd"`
- On Windows, command lines like `net start|stop "service"`
- If necessary, application with a procedure to recover after crash
- Remove automatic application start at boot and configure the boot start of the module instead

2.2.4 File replication prerequisites

- File directories that will be replicated are created on both servers
- They are located at the same place on both servers in the file tree
- It is better to synchronize clocks of both server for file replication (NTP protocol)
- On Linux, align uids/gids on both servers for owners of replicated directories/files
- See also system specific procedures and characteristics in section 2.1.6 page 28

2.3 Farm installation recommendation



2.3.1 Hardware prerequisites

- At least 2 servers with the same Operating System
- Supported OS: https://support.evidian.com/supported_versions/#safekit
- Linux: kernel compilation tools installed for vip kernel module

2.3.2 Network prerequisites

- 1 physical IP address per server (ip 1.1, ip 1.2, ip 1.3)
- If you need to set a virtual IP address (ip 1.20), servers must be in the same IP network with the standard SafeKit configuration (same LAN or extended LAN between remote computer rooms). For setting a virtual IP address with servers in different IP networks, see section 13.5.3 page 243.
- See also system specific procedures and characteristics in section 2.1.6 page 28

2.3.3 Application prerequisites

The same prerequisites as for a mirror module described in section 2.2.3 page 29

2.4 SafeKit upgrade

2.4.1 When proceed to an upgrade?

If you encounter a problem with SafeKit, see the Software Release Bulletin (English – HTML) on https://support.evidian.com/safekit containing the list of fixes on the product

If you want to take advantage of some new features, see the Release Notes on https://support.evidian.com/safekit. This document also tells you if you are in the case of a major upgrade (ex. 7.4 to 7.5) which requires a different procedure from the one presented here.

The upgrade procedure consists in uninstalling the old package and then installing the new package. All servers should be upgraded at the same time.

2.4.2 Prepare the upgrade

1. note the state "on" or "off" of services and modules started automatically at boot `safekit boot webstatus; safekit boot snmpstatus; safekit boot status -m AM` (where AM is the name of the module).

Important

Since SafeKit 7.5, the start at boot of the module can be defined in its configuration file. If so, the use of the `safekit boot` command becomes unnecessary.

2. for a mirror module

   note the server in the `ALONE` or `PRIM` status to know which server holds the up-to-date replicated files

3. optionally, take snapshots of modules

   Uninstalling/reinstalling will reset SafeKit logs and dumps of each module. If you want to keep this information (logs and last 3 dumps and configurations), run the command `safekit snapshot -m AM /path/snapshot_xx.zip` (replace AM by the module name)

2.4.3 Uninstall procedure

On Windows as administrator and on Linux as root:

1. stop all modules using the command `safekit shutdown`

   For a mirror in the `PRIM-SECOND` status, stop first the `SECOND` server to avoid an unnecessary failover

2. close all editors, file explorers, shells, or terminal under `SAFE` and `SAFEVAR` (to avoid package uninstallation error)

3. uninstall SafeKit package

   - in Windows, using the Control Panel-Add/Remove Programs applet
   - in Linux, using the command `safekit uninstall`

4. undo all configurations that you have done manually for the firewall setup (see section 10.3 page 156)

Uninstalling SafeKit includes creating a backup of the installed modules in `SAFE/Application_Modules/backup`, then unconfiguring them.

2.4.4 Reinstall procedure

1. install the new package as described in section 2.1 page 25

2. check with the command `safekit level` the installed SafeKit version and the validity of the license (which has not been uninstalled)

   If you have a problem with the new package and the old key, take a temporary license: see section 2.1.5 page 28

3. if you use the web console, clear the browser cache and refresh pages in the web browser

4. reconfigure all the installed modules

   web console/ Configuration/ ▼ on the module/ Edit the configuration/ or the command `safekit config -m AM` (replace AM by the module name)

5. if necessary, reconfigure the automatic start of modules at boot.

   Since SafeKit 7.5, the start at boot of the module can be defined in its configuration file. If so, skip this step. Otherwise, run web console/ Control/ on the node/Admin submenu/Configure boot start/ or command `safekit boot -m AM on` (replace AM by the module name)

Moreover, in the special cases:

- the automatic start at boot of the `safewebserver` service was disabled reconfigure it with the command `safekit boot weboff`

⇨ the automatic start at boot of the `safeagent` service was enabled

reconfigure it with the command `safekit boot snmpon`

⇨ `SAFE/web/conf/` or `SAFE/snmp/conf/snmpd.conf` files were customized

set back customizations in newly installed files (customizations have been saved in `SAFE/web/conf/<file name>.conf.<date>` and `SAFE/snmp/conf/snmpd.<date>`)

To restart modules after upgrade:

✓ farm module

web console/● Control/▾ on the module/▶ Start/ or command `safekit start -m AM` (replace AM by the module name)

✓ mirror module

On the server that has the up-to-date replicated files (former `PRIM` or `ALONE`): web console/● Control/▾ on the node/Expert/Force start/as prim/ or command `safekit prim -m AM` (replace AM by the module name)

On the other server (former `SECOND`): web console/● Control/▾ on the node/Expert/Force start/as second/ or command `safekit second -m AM` (replace AM by the module name)

2.5 SafeKit full uninstall

For completely removing the SafeKit package, follow the procedure described below.

2.5.1 On Windows as administrator

1. stop all modules using the command `safekit shutdown` (`SAFE=C:\safekit` if `%SYSTEMDRIVE%`=C:)
2. close all editors, file explorers, shells, or cmd under `SAFE` and `SAFEVAR` (to avoid package uninstallation error)
3. uninstall SafeKit using the Control Panel-Add/Remove Programs applet
4. reboot the server
5. delete the folder `SAFE` that is the installation directory of the previous install of SafeKit
6. undo all configurations that you have done for SafeKit boot/shutdown (see section 10.4 page 161)
7. undo all configurations that you have done for firewalls rules setting (see section 10.3 page 156)
8. if present, delete the user created by the previous install (default is SafeKitUser) with the command: `net user SafeKitUser /delete`

2.5.2 On Linux as root

1. stop all modules using the command `safekit shutdown` (`SAFE=/opt/safekit`)
2. uninstall SafeKit using the `safekit uninstall -all` command and answer `yes` when prompted to delete all SafeKit folders

3. reboot the server
4. undo all configurations that you have done for firewalls rules setting

   See section 10.3 page 156

2.6 SafeKit documentation

➩ SafeKit User's Guide (English, French - PDF and HTML)

It is this guide. Be sure to consult the guide corresponding to your SafeKit release number.

Available at https://support.evidian.com/safekit

➩ SafeKit Release Notes (English –PDF)

It presents:

✓ latest install instructions

✓ major changes

✓ restrictions and known problems

✓ migration instructions

Available at https://support.evidian.com/safekit

➩ Software Release Bulletin (English – HTML)

It details:

✓ up-to-date list of supported operating systems

✓ list of fixes and changes

Available at https://support.evidian.com/safekit

➩ SafeKit Knowledge Base (English – HTML)

It provides a selected list of KBs. Other KBs are available but require an account on https://support.evidian.com. See access to Evidian support described in section 8 page 133

Available at https://support.evidian.com/safekit

➩ SafeKit training

Available at https://www.evidian.com/products/high-availability-software-for-application-clustering/business-continuity-training-courses/

3. The SafeKit web console

Note

See the Release Notes, at https://support.evidian.com/safekit, for restrictions and known problems with the SafeKit web console.

3.1 Start the web console

Since SafeKit 7.5, by default, access to the web console requires the user to authenticate himself with a name and password. On SafeKit install, you had to initialize it with the user `admin` and assign a password. This `admin` name and password are sufficient to access all the console's features. For more details on this configuration, see 11.2.1 page 179.

3.1.1 Start a web browser

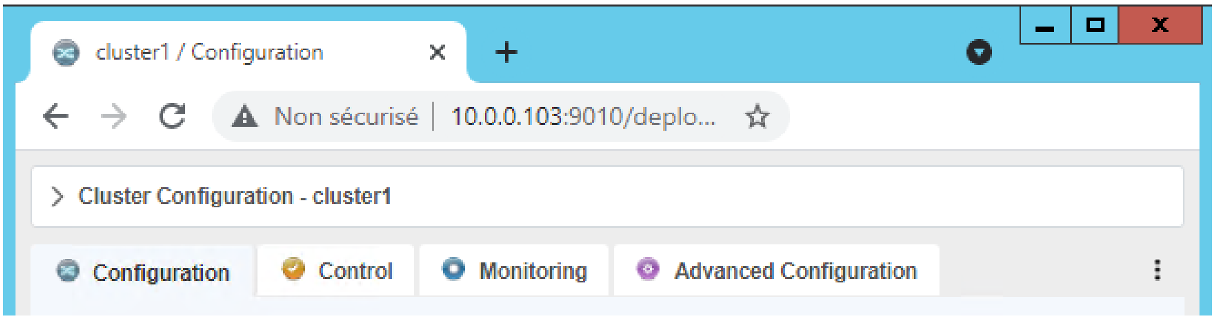
- The web browser runs on any allowed SafeKit nodes or workstation that can reach the SafeKit servers over the network.
- Network, firewall and proxy configuration must allow access to the administration network of all the servers that are administered with the web console
- JavaScript must be available and enabled in the web browser
- Tested browsers are Microsoft Edge, Firefox, and Chrome. The web console can also run-on smartphone and tablet. See the Release Notes on https://support.evidian.com/safekit for the supported versions of browsers
- To avoid security popups in Microsoft Edge, you may add the SafeKit servers addresses into the Intranet or Trusted zone
- The messages in the web console are displayed in French, English, Japanese languages, according to the preferred language configured into the web browser (for not supported languages, English is displayed). The message catalogs are in `SAFE/web/htdocs/jquery.lang/langpack/`.
- After SafeKit upgrade, you must clear the browser’s cache to get the new web console pages. A quick way to do this is a keyboard shortcut that works on IE, Firefox, and Chrome. Open the browser to any web page and hold `CTRL` and `SHIFT`

while tapping the `DELETE` key. The dialog box will open to clear the browser. Set it to clear everything and click Clear Now or Delete at the bottom. Close the browser, stop all background processes that may be still running and re-open it fresh to reload the web console.

3.1.2 Connect to a SafeKit server

The web console permits to administer one or more SafeKit clusters. A SafeKit cluster is a set of servers where SafeKit is installed and running. All servers belonging to a given SafeKit cluster share the same cluster configuration (list of servers and networks used) and communicate with each other's to have a global view of SafeKit modules configurations. The same server can not belong to many SafeKit clusters.

To manage a SafeKit cluster, connect to the default URL **`http://servername:9010`** (`servername` is the name or IP address of one of the nodes in the cluster). If HTTPS is configured, there is an automatic redirection to `https://servername:9453`.



The `localhost` and `127.0.0.1` values for `servername` are not allowed. If used, there is redirection on a page that requires the fill of another address or DNS name for the server.

The web console provides:

- ⇨ > Cluster Configuration panel: configure the servers that belong to the SafeKit cluster (see section 3.2 page 37)
- ⇨ Configuration tab: module quick installation and configuration on the cluster (see section 3.3 page 41)
- ⇨ Control tab: runtime administration of modules of the cluster (start/stop…) (see section 3.4 page 50)
- ⇨ Monitoring tab: status monitoring of modules of the cluster (see section 3.6 page 55)
- ⇨ Advanced Configuration tab: expert configuration and management of modules of the cluster (see section 3.7 page 56)
- ⇨ Cluster inventory panel. By default, there is only one managed cluster that is named `cluster1`. If you want to change this default name or manage another cluster with the same browser window, see section 3.10 page 66.

Note

The web console offers contextual help by clicking on the ⓘ icon.

3.2 Configure a SafeKit Cluster

The SafeKit cluster must be defined before installing, configuring, or starting a SafeKit module. A Safekit cluster is defined by a set of networks and the addresses, on these networks, of a group of SafeKit servers. These servers implement one or more modules. Each server is not necessarily connected to all the networks, but at least one.

With the Cluster Configuration panel, you can manage its configuration. In Control and Monitoring tabs, you can only display the current configuration of the SafeKit cluster. In Configuration and Advanced Configuration tabs, you can edit the configuration and apply it on all the nodes that belong to the cluster. The SafeKit cluster configuration is saved on the servers' side into the `cluster.xml` file (see section 12 page 227). For a correct behavior, it is required to apply the same cluster configuration on all the nodes. If you need to re-apply the configuration, switch to the Advanced edit mode, then click on the Apply button.

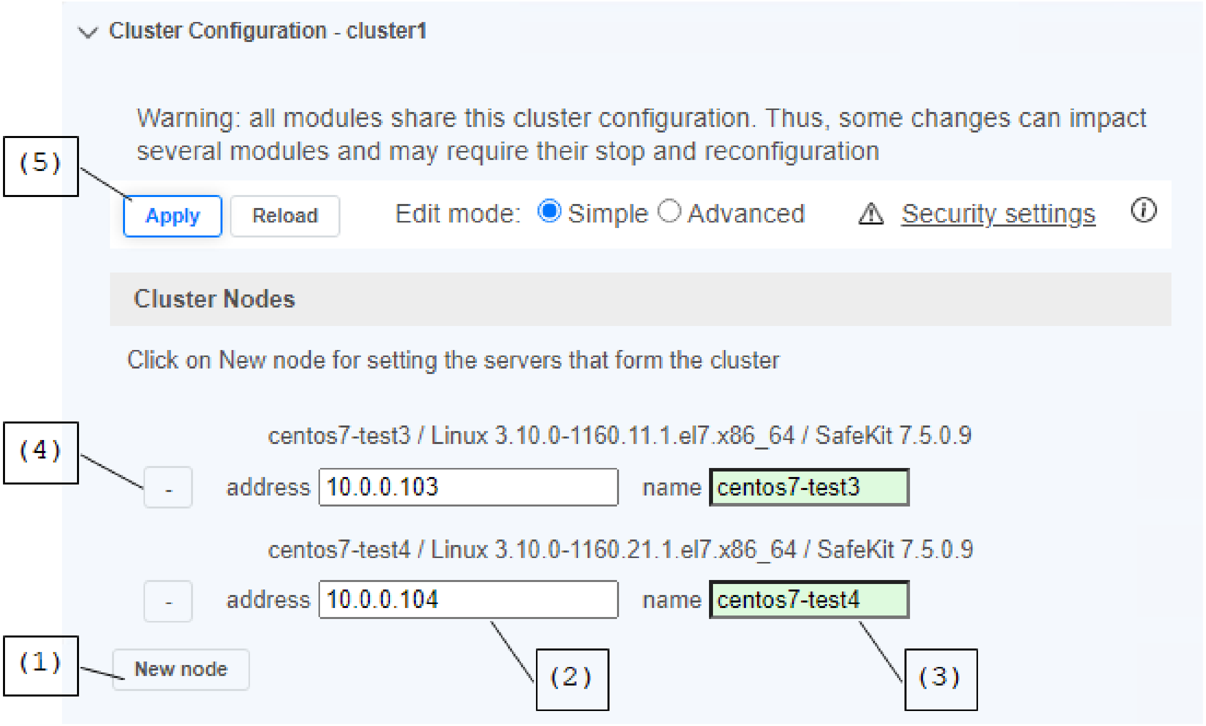
Important

You must fully define the SafeKit cluster configuration before installing and configuring modules since the modification of the SafeKit cluster can affect the configuration or the execution of installed modules.

3.2.1 Simple configuration

The simplest cluster configuration consists in defining all the nodes and their address on one network. For displaying or configuring the list of SafeKit cluster nodes:

- Click on Configuration or Advanced Configuration tabs
- Click on > Cluster Configuration to open the panel



Cluster Configuration - cluster1

Warning: all modules share this cluster configuration. Thus, some changes can impact several modules and may require their stop and reconfiguration

Edit mode: ● Simple ○ Advanced

Cluster Nodes

Click on New node for setting the servers that form the cluster

centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9

address 10.0.0.103 name centos7-test3

centos7-test4 / Linux 3.10.0-1160.21.1.el7.x86_64 / SafeKit 7.5.0.9

address 10.0.0.104 name centos7-test4

➪ (1) Click on the New node button for adding a new node into the cluster

➪ (2) Fill in the administration IP address of the node and then press the Tab key to check the server connectivity and automatically insert the server hostname. The protocol and port used to access the node is the same as the one used to connect to the SafeKit web service.

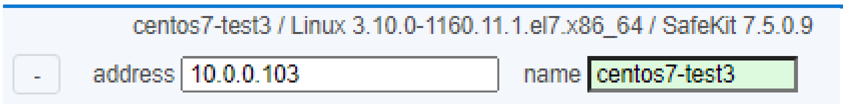
Important

Do not use `localhost` or `127.0.0.1` as IP address.

The server connected to the SafeKit web console must be included into the SafeKit cluster.

➪ (3) Change the node name if necessary. This name is the one that will be used by the SafeKit administration service for uniquely identifying a SafeKit server. It is also the one displayed into the SafeKit web console.

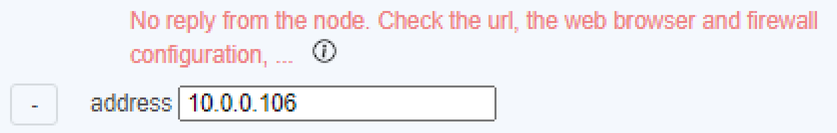
The name field background color reflects the reachability of the node.



centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9

- address 10.0.0.103 name 🟩 centos7-test3

Green color means that the SafeKit server is available.



No reply from the node. Check the url, the web browser and firewall configuration, ... ⓘ

- address 10.0.0.106

Red color means that the web console had no reply from the server within the timeout delay. Fix the problem to be able to administer this node. It may be a bad address, a network or host failure, a bad configuration of the web browser or the firewall, the stop of the SafeKit web service on the node. For solving the problem, refer to the section 7.1 page 109.

➪ (4) Click on the – button for removing the node from the SafeKit cluster.

Important

When removing a node from the cluster, all the modules installed on this node will not be any more usable.

➪ (5) Once the configuration is completed, click on the Apply button for saving changes and applying the configuration on all nodes. Click on the Reload button to discard changes and reload the initial configuration. When the Apply button becomes blue Apply, it means that some changes have been done and must be applied.

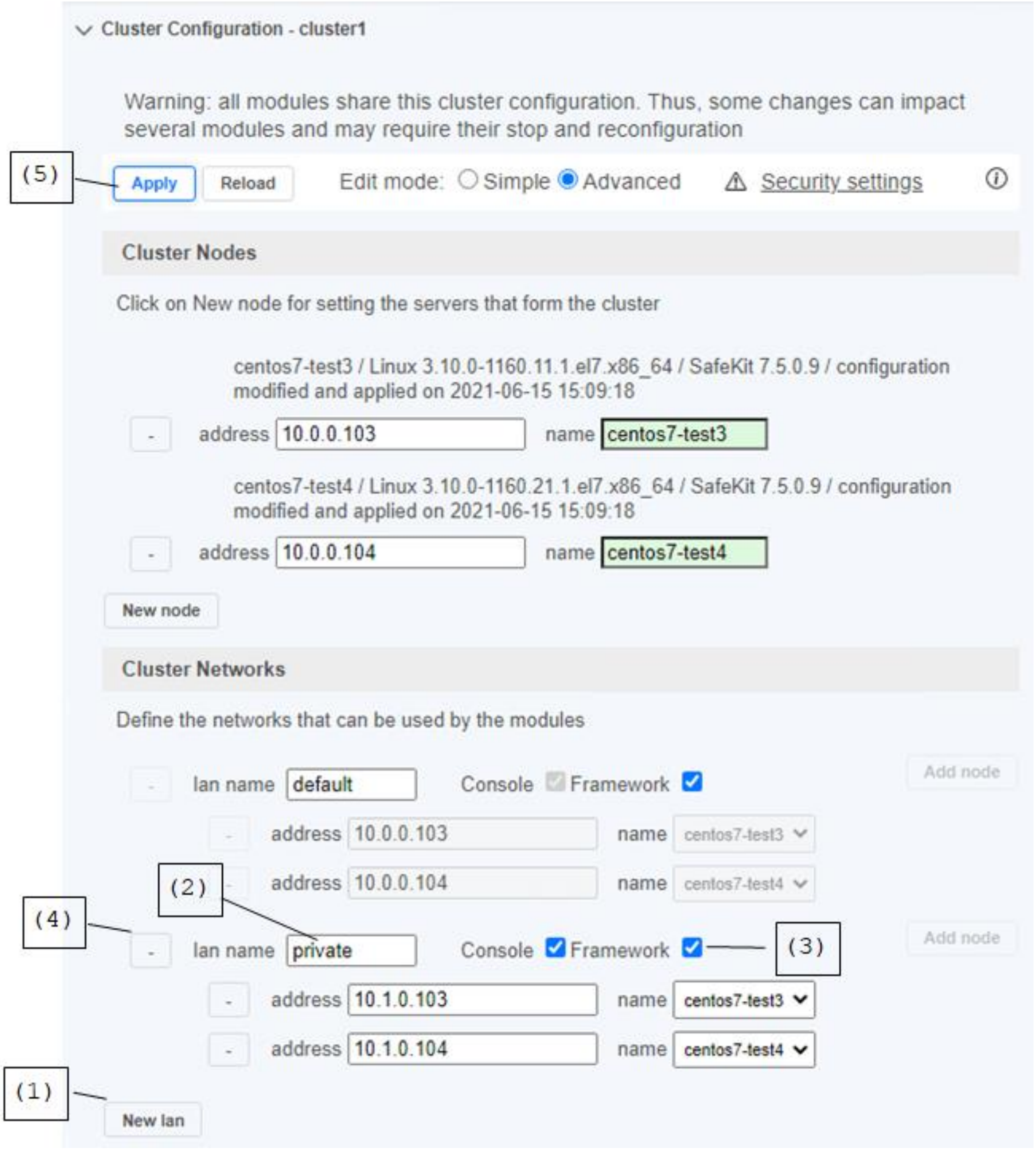
➪ (6) Click on ⌄ Cluster Configuration to close the panel.

3.2.2 Advanced configuration

With Simple edit mode, you can define only one network, that is the one used to connect the web console.

You can define more networks, for communication redundancy, by switching to the Advanced edit mode. The node name is used to identify the various IP addresses for the same node. The lan name provides the network topology abstraction and is used for configuring the networks used by a module.

- Click on Configuration or Advanced Configuration tabs
- Click on > Cluster Configuration to open the panel
- Click on the Advanced radio button
- The Cluster Nodes panel is the same as the one in simple edit mode



Cluster Configuration - cluster1

Warning: all modules share this cluster configuration. Thus, some changes can impact several modules and may require their stop and reconfiguration

Edit mode: ○ Simple ● Advanced

Cluster Nodes

Click on New node for setting the servers that form the cluster

centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9 / configuration modified and applied on 2021-06-15 15:09:18

address 10.0.0.103 name centos7-test3

centos7-test4 / Linux 3.10.0-1160.21.1.el7.x86_64 / SafeKit 7.5.0.9 / configuration modified and applied on 2021-06-15 15:09:18

address 10.0.0.104 name centos7-test4

Cluster Networks

Define the networks that can be used by the modules

lan name default Console ☑ Framework ☑

address 10.0.0.103 name centos7-test3

address 10.0.0.104 name centos7-test4

lan name private Console ☑ Framework ☑

address 10.1.0.103 name centos7-test3

address 10.1.0.104 name centos7-test4

- The Cluster Networks panel displays the network that the web console connected to and then the additional networks

- (1) Click on the New lan button for adding a new network

- (2) Fill in a friendly name for the new network and set the IP address for the SafeKit cluster nodes on the new network. The lan name is used for configuring networks used by a module (see section 3.3.2.2 page 47).

- (3) Check or uncheck the boxes to specify the network use:
  - ✓ a Framework network is a network used for internal communications within the cluster. These are global cluster and module internal communications, as well as communications for executing distributed commands. You must define at least one framework network that includes all nodes in the cluster. It is recommended to define several framework networks to tolerate at least one network failure.
  - ✓ a Console network is a network on which the web console can connect for communicating with cluster nodes. This network must include all the cluster nodes. You can define multiple console networks according to administrative requirements and network topology.

- (4) Click on the – button for removing the network

Important

When removing a lan, all the modules configured for using this lan will need to be stopped and reconfigured.

- (5) Once the configuration is completed, click on the Apply button for saving and applying the configuration on all nodes. Click on the Reload button to discard changes and reload the initial configuration. When the Apply button gets blue Apply, it means that some changes have been done.

- Click on ⌄ Cluster Configuration to close the panel

3.2.3 Configuration with command line

The command line equivalent to the SafeKit cluster configuration is described below. Replace `node1` and `node2` by the name of your cluster nodes set into the SafeKit cluster configuration.

1. Log as administrator/root and open a command shell window on one node
2. Edit the file `SAFEVAR/cluster/cluster.xml`

   `SAFEVAR` is `C:\safekit\var` on Windows if `%SYSTEMDRIVE%`=C:, `/var/safekit` on Linux

The file content is for instance:

```
<?xml version="1.0" encoding="utf-8"?>
<cluster>
<lans>
   <lan name="default" console="on" framework="on">
      <node name="node1" addr="10.0.0.103"/>
      <node name="node2" addr="10.0.0.104"/>
   </lan>
   <lan name="private" console="on" framework="on">
      <node name="node1" addr="10.1.0.103"/>
      <node name="node2" addr="10.1.0.104"/>
   </lan>
</lans>
</cluster>
```

3. Run `safekit cluster config`

   To locally apply the cluster configuration in `SAFEVAR/cluster/cluster.xml`

4. Run `safekit -H "*" -G`

   to apply the local configuration on all SafeKit nodes defined into cluster.xml

For more details, refer to section 12 page 227.

3.3 Configure a module

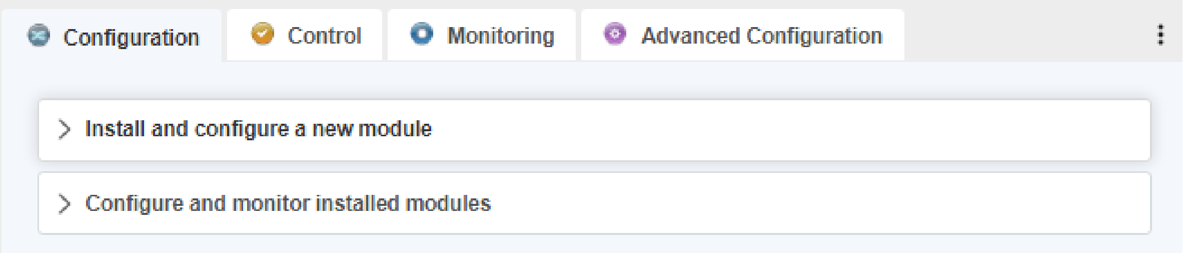
This tab offers, thanks to the Configuration wizard, the quick installation and configuration of a new module or the quick reconfiguration of a module already installed.

Important

The Configuration wizard allows setting only major parameters and user scripts edition. If you need to edit other `userconfig.xml` fields or advanced management, complete the quick configuration then go to the Advanced Configuration tab (see section 3.7.1 page 58).

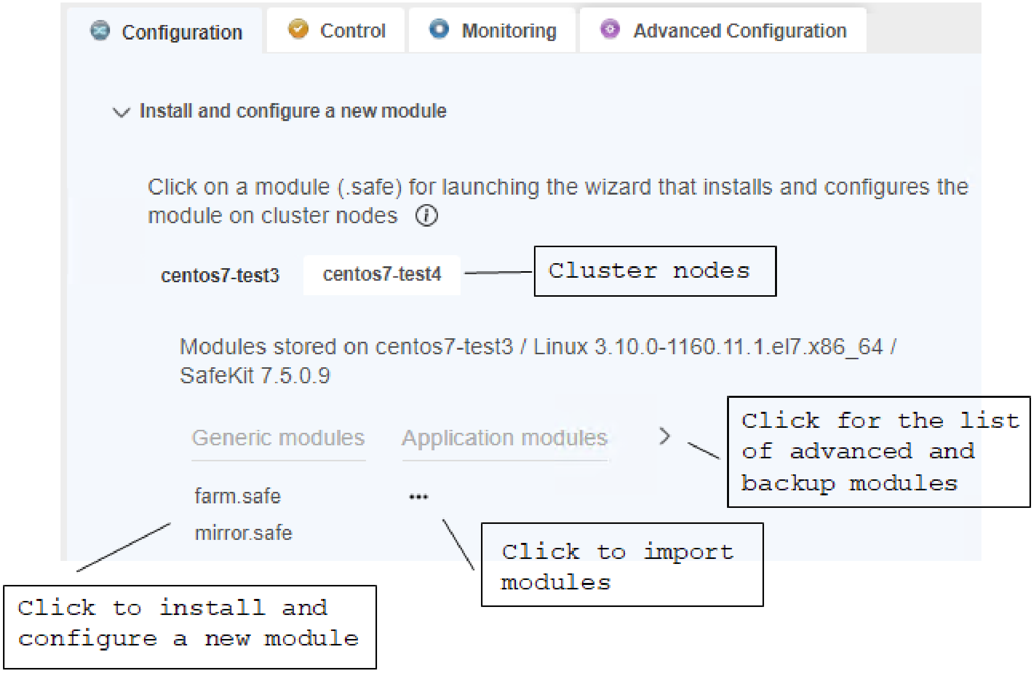
When reconfiguring an installed module, the Configuration wizard forces to apply the configuration on all the nodes of the module. If you need to apply it only on a subset, use instead the Advanced configuration wizard available in the Advanced Configuration tab.

⇨ In Configuration tab


Configuration
Control
Monitoring
Advanced Configuration
> Install and configure a new module
> Configure and monitor installed modules

5. › Install and configure a new module

Click on › to open the panel for installing and configuring a new module.



⌄ Install and configure a new module

Click on a module (.safe) for launching the wizard that installs and configures the module on cluster nodes ⓘ

centos7-test3 centos7-test4

Modules stored on centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9

farm.safe
mirror.safe

For each SafeKit cluster node, it lists the modules templates that are stored on the server.

✓ Generic modules

Lists the generic modules `mirror.safe` and `farm.safe`. Use a generic module for integrating a new application based on a mirror or farm architecture. These modules are stored in the `SAFE/Application_Modules/generic` directory on the server side.

✓ Application modules

Lists all the modules customized for a specific business application. These modules are stored into `SAFE/Application_Modules/demo` (that may not exist if empty)

✓ Advanced modules

Lists all advanced modules for advanced integration. These modules are stored into `SAFE/Application_Modules/other`.

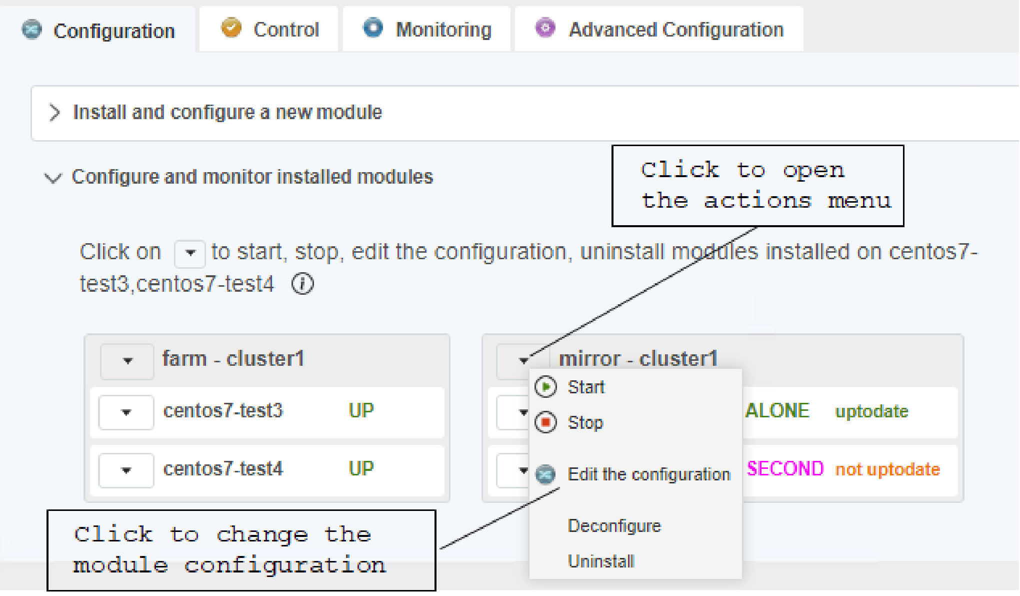
✓ Backup modules

Lists all the modules stored into `SAFE/Application_Modules/backup`. This directory is used for saving module's configuration on module uninstall.

You can make your module template available from Configuration tab by copying the template into one of these areas (see section 3.8 page 63).

6. › Configure and monitor installed modules

   Click on › to open the panel for re-configuring and monitoring installed modules.


Configuration
Control
Monitoring
Advanced Configuration
› Install and configure a new module
⌄ Configure and monitor installed modules
Click to open the actions menu
Click on ▾ to start, stop, edit the configuration, uninstall modules installed on centos7-test3,centos7-test4 ⓘ
farm - cluster1
centos7-test3 UP
centos7-test4 UP
mirror - cluster1
Start
Stop
Edit the configuration
Deconfigure
Uninstall
ALONE uptodate
SECOND not uptodate
Click to change the module configuration

This panel displays the modules installed (under the `SAFE/modules` directory) on all the nodes set into the SafeKit cluster (see section 3.2 page 37), and their current state. The list is empty after a fresh SafeKit install. Installed modules can be re-configured to modify general parameters for instance, started, stopped, or uninstalled.

3.3.1 Select the module to configure

- In Configuration tab
- › Install and configure a new module

  Click on the name of the module to configure among the generic, application, advanced or backup modules. You first have to set the new module name. Then click on the - button to open the Configuration wizard.
- › Configure and monitor installed modules

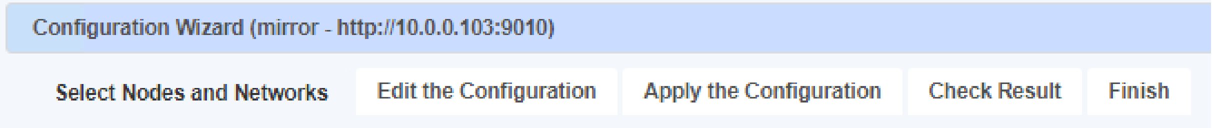
  For installed modules, click on the ▾ button (next to the module name) for opening the actions menu and choose Edit the configuration. First, select the node that you want to use for editing the configuration (this node is called the source server). Then click on the Confirm button to open the Configuration wizard that offers the configuration files edition from the selected node.

The configuration wizard is a multi-step interface that helps you configure and install a module on one or many servers of your cluster. You just have to fill intended parameters and click on the button to go to the next step.

For a first test of SafeKit, configure a new generic module `mirror.safe` or `farm.safe`.

3.3.2 Configuration wizard

The Configuration wizard is (example of the new module configuration from mirror.safe template):


Configuration Wizard (mirror - http://10.0.0.103:9010)
Select Nodes and Networks
Edit the Configuration
Apply the Configuration
Check Result
Finish

The wizard walks you through the process of configuring a module:

- 3.3.2.1 "Select nodes and networks for the module" page 44

  This tab is used to set the nodes on which the module is configured. It also defines the networks for the module.

- 3.3.2.2 "Edit the configuration of the module" page 47

  This tab allows setting only major parameters for quick configuration of the module.

- 3.3.2.3 "Apply the configuration of the module" page 48

  In this tab, you can apply the changes to take effect. It forces to apply the configuration on all the nodes (set into the first tab) and requires that the module is stopped on all of them.

- 3.3.2.4 "Check the configuration result" page 49

  This tab shows the result of the previous step.

- 3.3.2.5 "Finish" page 49

Important

If you close the Configuration wizard before applying the configuration, it aborts the module configuration. But if you have made changes in previous tabs and have validated, these changes have been saved on the source server.

Once you click on the button to apply the configuration, this one cannot be cancelled.

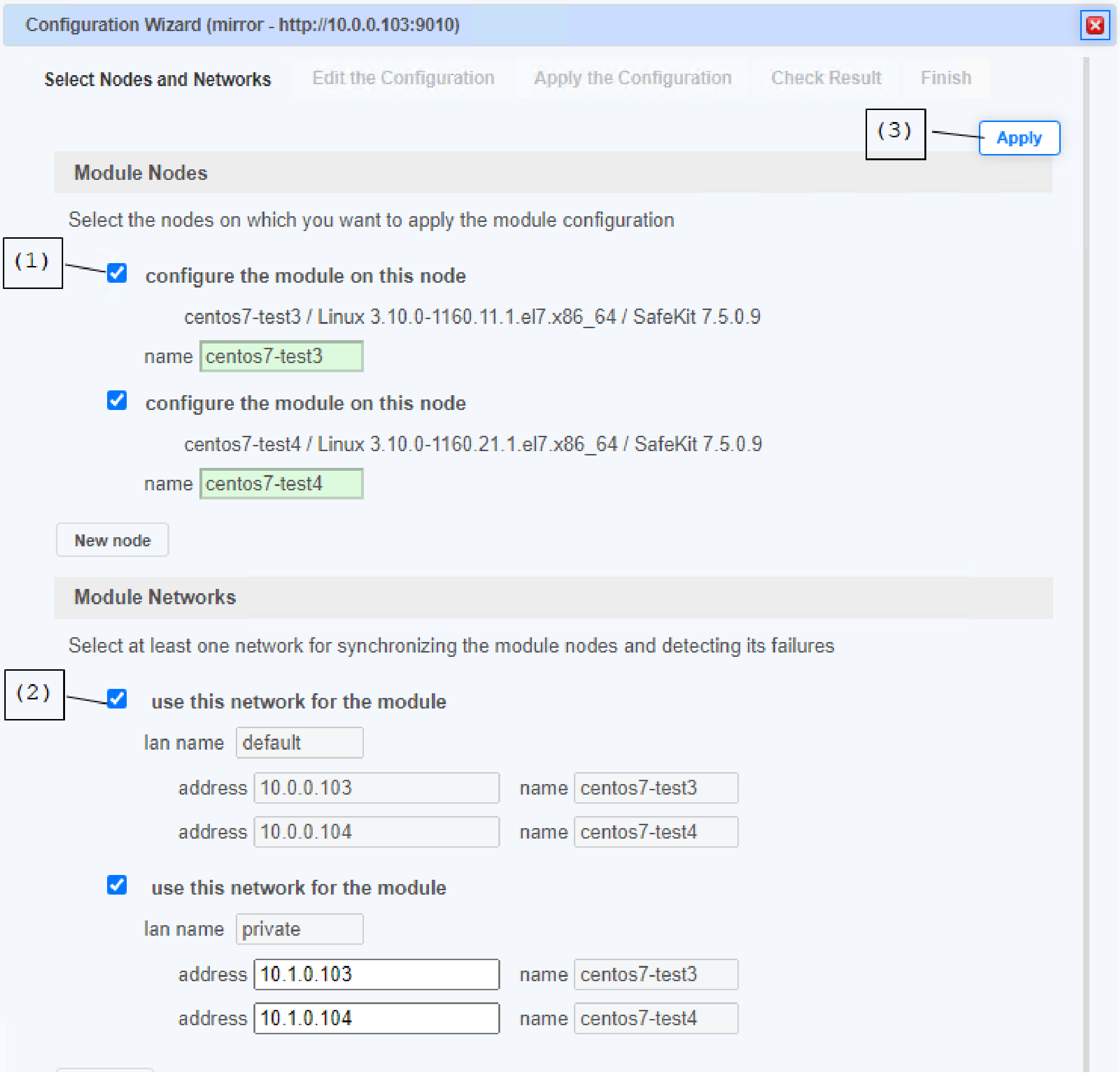
3.3.2.1 Select nodes and networks for the module

- In Configuration wizard
- Select Nodes and Networks tab

  This form is used to select the nodes on which the module will be configured and to define the networks used by the module.

Note

For old module templates, the networks selection panel is not available.



Configuration Wizard (mirror - http://10.0.0.103:9010)

Module Nodes

Select the nodes on which you want to apply the module configuration

☑ configure the module on this node

centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9

name centos7-test3

☑ configure the module on this node

centos7-test4 / Linux 3.10.0-1160.21.1.el7.x86_64 / SafeKit 7.5.0.9

name centos7-test4

Module Networks

Select at least one network for synchronizing the module nodes and detecting its failures

☑ use this network for the module

lan name default

address 10.0.0.103 name centos7-test3

address 10.0.0.104 name centos7-test4

☑ use this network for the module

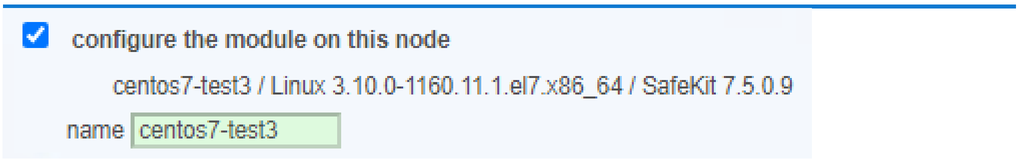
lan name private

address 10.1.0.103 name centos7-test3

address 10.1.0.104 name centos7-test4

⇨ Check the box for selecting the nodes that implement the module.

The name field color reflects the reachability of the node.

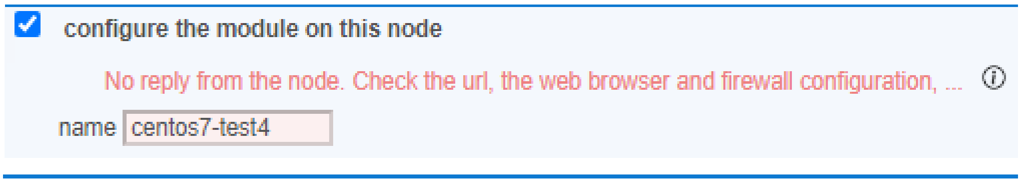


☑ configure the module on this node

centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9

name centos7-test3

Green color means that the SafeKit server is available.



☑ configure the module on this node

No reply from the node. Check the url, the web browser and firewall configuration, ... ⓘ

name centos7-test4

Red color means that the web console had no reply from the server within the timeout delay.

You can either choose to:

✓ Fix the problem to be able to configure this node. It may be a bad address, a network or host failure, a bad configuration of the web browser or the firewall, the stop of the SafeKit web service on the node…

✓ Uncheck the box to not configure it

✓ Keep the node checked (if you think the node is temporarily unreachable).

⇨ Add new node(s) when necessary (2 nodes for mirror architecture, at least 2 nodes for farm architecture). This is a shortcut for the SafeKit cluster configuration (see section 3.2 page 37).

⇨ (2) Check the box for selecting the networks used by the module. Select at least one network for synchronizing the module nodes and detecting its failures. But it is strongly recommended to set two monitoring networks to avoid the split-brain case.

The lan name is the one used in next tab for the configuration of the module.

⇨ Add new monitoring network when necessary and check the box for using this network for the module. This is a shortcut for the SafeKit cluster configuration (see section 3.2 page 37).

⇨ (3) Click on the Apply button to save changes and go to the next step

Important

Node list or network list modification is equivalent to modifying the SafeKit cluster configuration. In that case, the Apply button apply the changes on all the SafeKit cluster nodes.

For a first test of SafeKit:

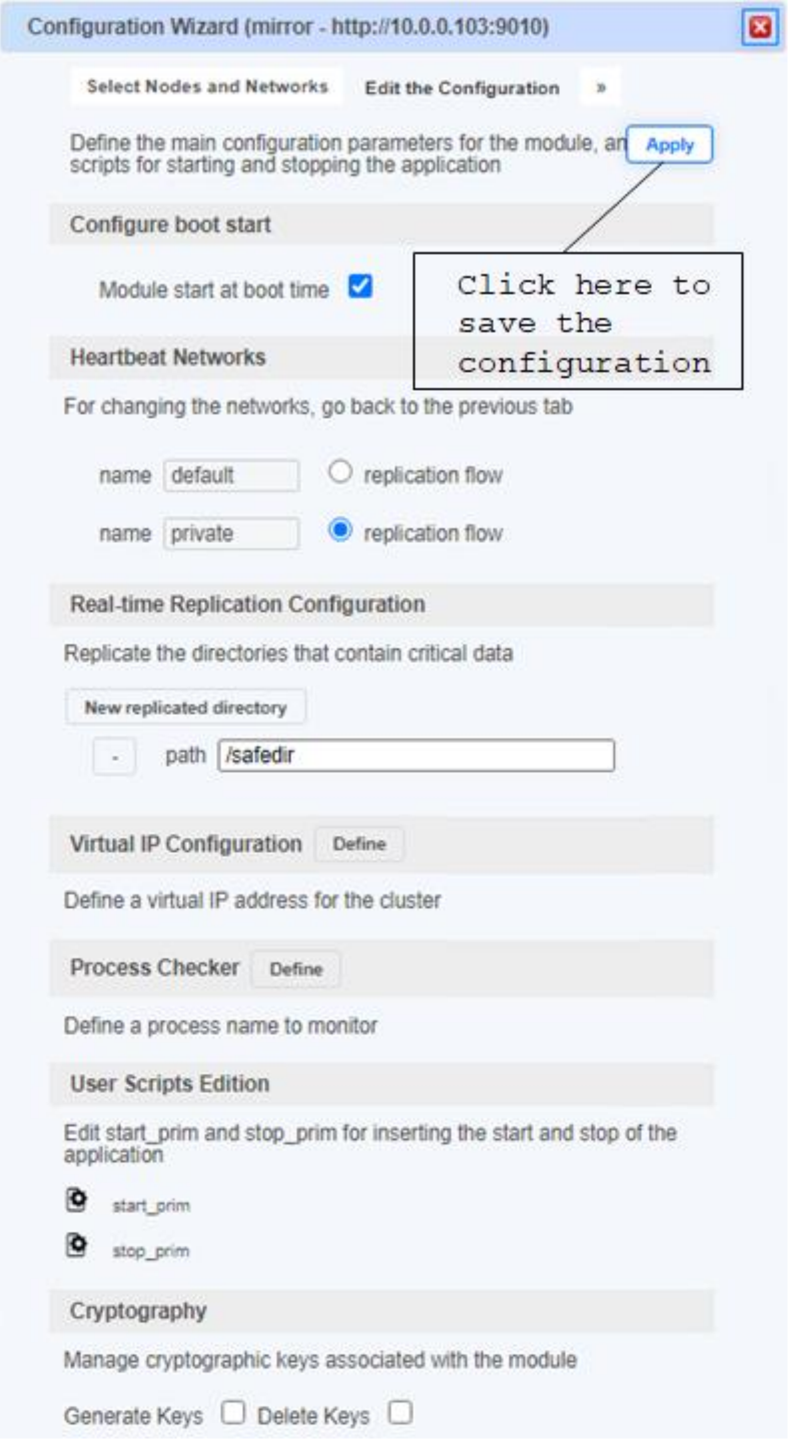
⇨ Follow this procedure to define the cluster nodes and monitoring networks for the module

3.3.2.2 Edit the configuration of the module

This form allows setting only major parameters for quick configuration of the module. If you need to edit other `userconfig.xml` fields or advanced configuration, complete this configuration then go to the Advanced Configuration tab (see section 3.7.1 page 58).

- ⇨ In Configuration wizard
- ⇨ Edit the configuration tab
- ⇨ Fill in the form and edit scripts
- ⇨ You can also secure SafeKit communication between cluster nodes of the module by managing cryptographic keys associated with the module (for details, see section 10.5 page 162).
- ⇨ Once the edition is completed, click on the Apply button to save changes and go to the next step

Note that the heartbeat networks are defined with network name set into the previous tab. For module templates before SafeKit 7.2, you must fill the IP address for each node.



Configuration Wizard (mirror - http://10.0.0.103:9010)

Define the main configuration parameters for the module, an scripts for starting and stopping the application

Click here to save the configuration

Configure boot start

Module start at boot time ☑

Heartbeat Networks

For changing the networks, go back to the previous tab

name default ○ replication flow

name private ● replication flow

Real-time Replication Configuration

Replicate the directories that contain critical data

path /safedir

Virtual IP Configuration

Define a virtual IP address for the cluster

Process Checker

Define a process name to monitor

User Scripts Edition

Edit start_prim and stop_prim for inserting the start and stop of the application

Cryptography

Manage cryptographic keys associated with the module

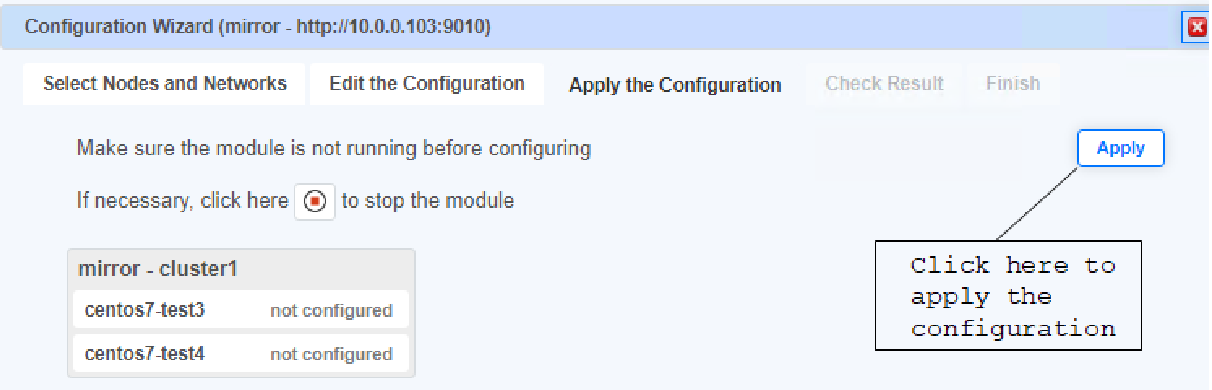
Generate Keys □ Delete Keys □

For a first test of SafeKit:

- Follow this procedure to configure mirror or farm
- Get familiar with the module control and monitoring before inserting application specific start and stop commands into user scripts

3.3.2.3 Apply the configuration of the module

- In Configuration wizard
- Apply the Configuration tab


Configuration Wizard (mirror - http://10.0.0.103:9010)
Select Nodes and Networks
Edit the Configuration
Apply the Configuration
Check Result
Finish
Apply

Make sure the module is not running before configuring

If necessary, click here ⊙ to stop the module

mirror - cluster1	
centos7-test3	not configured
centos7-test4	not configured

Click here to apply the configuration

- Check the module state for the cluster. When “not configured”, go to (3).

- If the module in not in the `STOP` (red) state, click on the ⊙ button to stop the module and then wait for `STOP` (red) state on all nodes before going to (3). The configuration will be allowed only when the module is stopped on all nodes.

- Click on the Apply button to apply the configuration on all the nodes

  The configuration may take some time to run commands on all the cluster nodes. Once finished, the Check Result tab is enabled.

Note

If you do not want to apply the configuration on all nodes, use instead the Advanced configuration wizard available in the ⚙ Advanced Configuration tab.

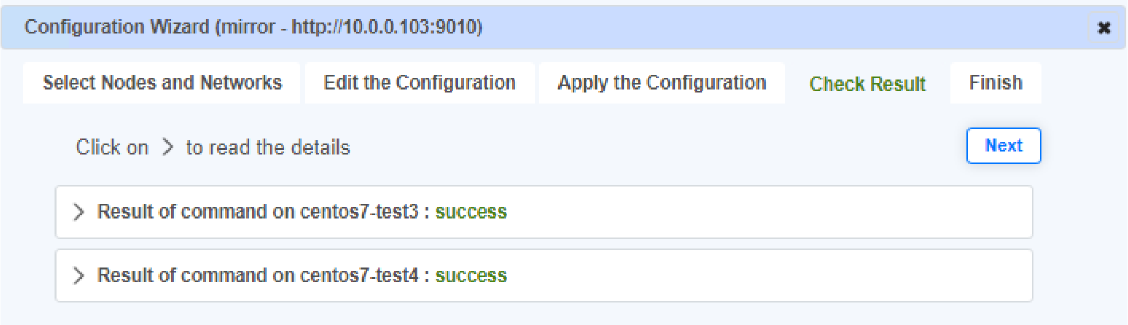
Important

When reconfiguring installed modules, the entire module configuration directory `SAFE/modules/AM` is deleted (where AM is the module name) and rebuild it from the changes made in the console: on the server side, close all editors, file explorers, shells or cmd under `SAFE/modules/AM` before configuring (otherwise there is a risk that the deployment of the new module files goes wrong).

3.3.2.4 Check the configuration result

➾ In Configuration wizard

➾ Check Result tab



Configuration Wizard (mirror - http://10.0.0.103:9010)
✖
Select Nodes and Networks
Edit the Configuration
Apply the Configuration
Check Result
Finish
Next

Click on > to read the details

> Result of command on centos7-test3 : success

> Result of command on centos7-test4 : success

This tab is red if the configuration has failed on one or more nodes. It is green if the configuration is successful on all nodes.

➾ Read the configuration result for each node:

- ✓ success means that configuration is successful on the node.
- ✓ connection error signals a connection failure with the node. Once the connectivity problem is fixed, you can go back to the Apply the Configuration tab and Apply again.
- ✓ failure is displayed when one of the server-side commands run during configuration has failed. Click on Result of command to read the output of the commands and find out the error. You may need to change parameters or connect to the server to address the problem. Once the problem is fixed, go back to the Apply the Configuration tab, and Apply again.

➾ Click on the Next button or close the window to dismiss the Configuration wizard.

3.3.2.5 Finish

➾ In Configuration wizard

➾ Finish tab

This tab ends the configuration wizard. Its main interest is for the first configuration and start of a mirror module with replicated directory. In that case, it proposes to select the server that has got the up-to-date replicated directories and to start it as primary (see section 5.3 page 97).

3.3.3 Configuration with command line

The command line equivalent to the Configuration wizard is described below. Replace `AM` by your module name; replace `node1` and `node2` by the name of your cluster nodes set during the SafeKit cluster configuration.

1. Log as administrator/root and open a command shell window on one node

   For instance, log-in `node1`

2. Run `safekit module install -m AM SAFE/Application_Modules/generic/mirror.safe`

   to install a new module named `AM`, from `mirror.safe` template

3. Edit the module configuration and scripts in `SAFE/modules/AM/conf` and `SAFE/modules/AM/bin`

4. Run `safekit module genkey -m AM` or `safekit module delkey -m AM`

   to create or delete cryptographic key for the module

5. Run `safekit -H "node1,node2" -E AM`

   to (re)install the module AM and apply its configuration, that is get from the node running the command (`node1` in this example). It applies it on all listed nodes (`node1` and `node2).`

For more details on commands, refer to section 9.6 page 149.

3.4 Control a module

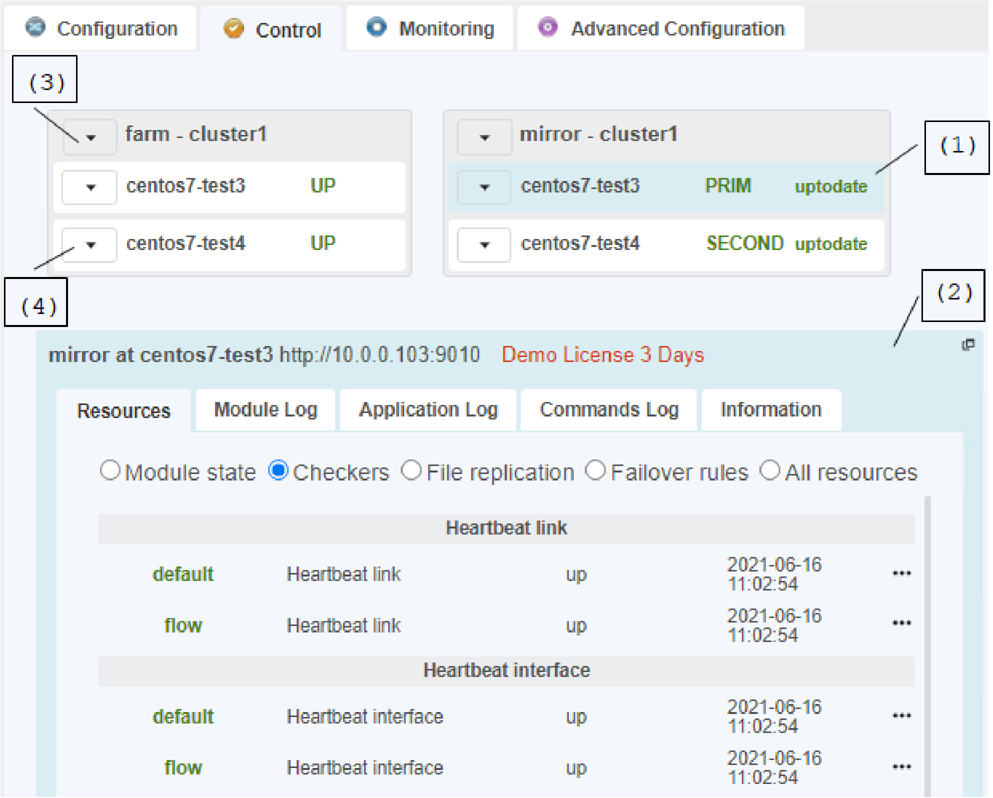
3.4.1 Select module and node

⇨ In ✓ Control tab

By default, modules that can be controlled are the modules installed on all the nodes from the SafeKit cluster.

For each module:

- ✓ the name of installed modules and of the cluster
- ✓ the current state of the module instances on all cluster nodes are displayed
- ✓ you can run a global action on all nodes in the cluster or local action only on one node, and view the detailed status
- ✓ the detailed state for the selected module/node



farm - cluster1	
centos7-test3	UP
centos7-test4	UP

mirror - cluster1		
centos7-test3	PRIM	uptodate
centos7-test4	SECOND	uptodate

mirror at centos7-test3 http://10.0.0.103:9010 Demo License 3 Days

○ Module state ● Checkers ○ File replication ○ Failover rules ○ All resources

Heartbeat link				
default	Heartbeat link	up	2021-06-16 11:02:54	···
flow	Heartbeat link	up	2021-06-16 11:02:54	···
Heartbeat interface				
default	Heartbeat interface	up	2021-06-16 11:02:54	···
flow	Heartbeat interface	up	2021-06-16 11:02:54	···

➾ (1) For viewing the detailed status of the module on a given node, click on the node

All nodes set during module configuration are listed under the module. The selected node is highlighted with a blue color.

➾ (2) Look at the panel, with the blue background color, to check the status on the selected module/node

- ✓ select the Resources tab to view the current resources status of the module. Place the mouse cursor over the resource name to get the internal name of the resource. Resource's state is controlled by the failover machine to trigger a failover on failures (see section 13.18 page 288). Click on ··· to display the value of the resource over time. This history may be empty for some resources (unassigned or cleaned).

  Note

  Since SafeKit 7.5, the date displayed is the last date the resource was assigned. Before SafeKit 7.5, this is the first time the resource has been assigned to the current value.

- ✓ select the Module Log tab to read the execution log of the module. Set or clear the verbose log's checkbox to display the short log (with only E messages) or the verbose log (all messages including debug ones); See also the troubleshooting section 7 page 109 for messages examples.

- ✓ select the Application Log tab to read application output messages of start and stop scripts. These messages are saved on the server side in `SAFEVAR/modules/AM/userlog.ulog` (where `AM` is the module name).

- ✓ select the Commands Log tab to display the `safekit` commands that have been executed on the node (commands applied on the module and all global commands).

- ✓ select the Informations tab to check the server level and the module configuration. It is the active configuration that is the last configuration successfully applied.

Note

In Module Log, Application Log and Commands Log tabs, click on the ⟳ button to get the last messages or on the ⤓ button to locally save the log.

- ➾ If you prefer, you can click on the ⧉ icon to display the detailed status into a new window

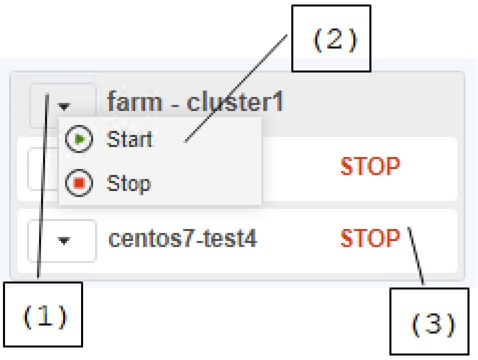
- ➾ (3) Click on the ▾ button of the module. It opens a menu for running a global ▶ Start or ■ Stop of the module on all the nodes in the cluster

- ➾ (4) Click on the ▾ button of one node. It opens a menu with all actions that will be executed only on the selected node. It includes local ▶ Start or ■ Stop of the module and many commands to control, monitor, and support the module on the node.

3.4.2 Control a farm module

For a first test of SafeKit on the farm module:

- ➾ (1) Click on the ▾ button of the module.
- ➾ (2) Select the global ▶ Start or ■ Stop. A confirmation dialog appears before running the command. The command result is displayed after completion only if an error occurs.
- ➾ (3) Wait for expected module state.
- ➾ On unexpected state, look at the detailed status of the node.



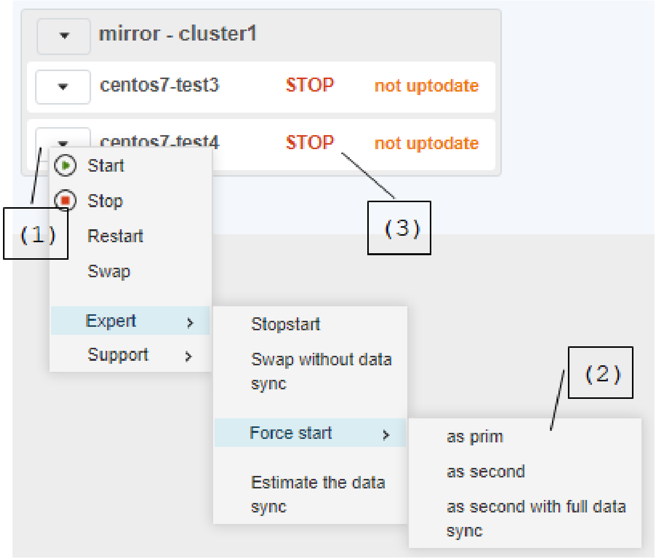
Refer to sections listed below:

- ➾ To continue the tests, see 4 Tests page 69
- ➾ To understand and check the correct behavior of a farm module, see section 6 page 105

3.4.3 Control a mirror module

For the first start of the mirror module, you cannot use the global start of the cluster. Instead, you must:

- ➾ (1) Click on the ▾ button of the node.
- ➾ (2) Select Expert/Force start as prim, if the node has the up-to-date replicated directories; otherwise, select Expert/Force start as second. A confirmation dialog appears before running the command. The command result is displayed after completion only if an error occurs.
- ➾ (3) Wait for expected module state.
- ➾ On unexpected state, look at the detailed state of the node.



Refer to sections listed below:

- ➾ For the first start-up of a mirror module, see section 5.3 page 97
- ➾ For the start-up of a mirror module with the up-to-date data, see section 5.5 page 99
- ➾ To continue the tests, see 4 Tests page 69
- ➾ To understand and check the correct behavior of a mirror module, see section 5 page 95

3.4.4 Control with command line

The command line equivalent to the module start is described below. Replace `AM` by your module name; replace `node1` and `node2` by the name of your cluster nodes set during the SafeKit cluster configuration.

For the global start:

1. Log as administrator/root and open a command shell window on one node

   For instance, log-in `node1`

2. Run `safekit -H "node1,node2" start -m AM`

   To start the module `AM` on all listed nodes (`node1` and `node2`). For a mirror module, it will start as primary or secondary according to the last module state.

For the local start:

1. Log as administrator/root and open a command shell window on one node

   For instance, log-in `node1`

2. Run `safekit start -m AM` or `safekit prim -m AM` or `safekit second -m AM`

   To start the module `AM` locally (that is on `node1`).

For a mirror module, use `prim` for starting it as primary; `second` for starting it as secondary. When using `start`, it will start as primary or secondary according to the last module state

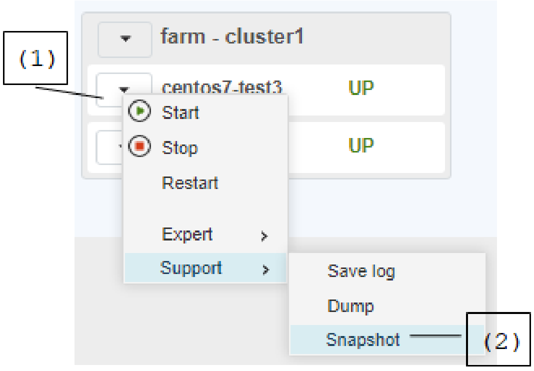
All control and monitor commands are detailed in sections 9.4 page 146 and 9.5 page 148.

3.5 Snapshots of module for support

When the problem is not easily identifiable, it is recommended to take a snapshot of the module on all nodes as described below. Snapshots allows an offline and in-depth analysis of the module and node status as described in section 7.16 page 123. If this analysis fails, send snapshots to support as described in section 8 page 133.

3.5.1 Snapshot of module

- ➩ In Configuration, Control, Monitoring, or Advanced Configuration tab
- ➩ Choose the module and the node



- ✓ (1) Click on the ▾ button of the node. It opens a menu with all actions that can be executed on the selected node.
- ✓ (2) Select the Support submenu, then Snapshot command. The Web console relies on the web browser download settings for saving the snapshot file on your workstation.

- ➩ Repeat this operation for the other nodes in the cluster.

3.5.2 Snapshot with command line

The command line equivalent to the module snapshot is described below. Replace `AM` by your module name.

1. Log as administrator/root and open a command shell window on one node

   For instance, log-in `node1`

2. Run `safekit snapshot -m AM /tmp/snapshot_xx.zip`

   To save the snapshot of the `AM` module in `/tmp/snapshot_xx.zip` (absolute path mandatory) locally (that is on `node1)` .

- ➩ Repeat all these commands on the other nodes in the cluster

For more details on support commands, refer to section 9.7 page 151.

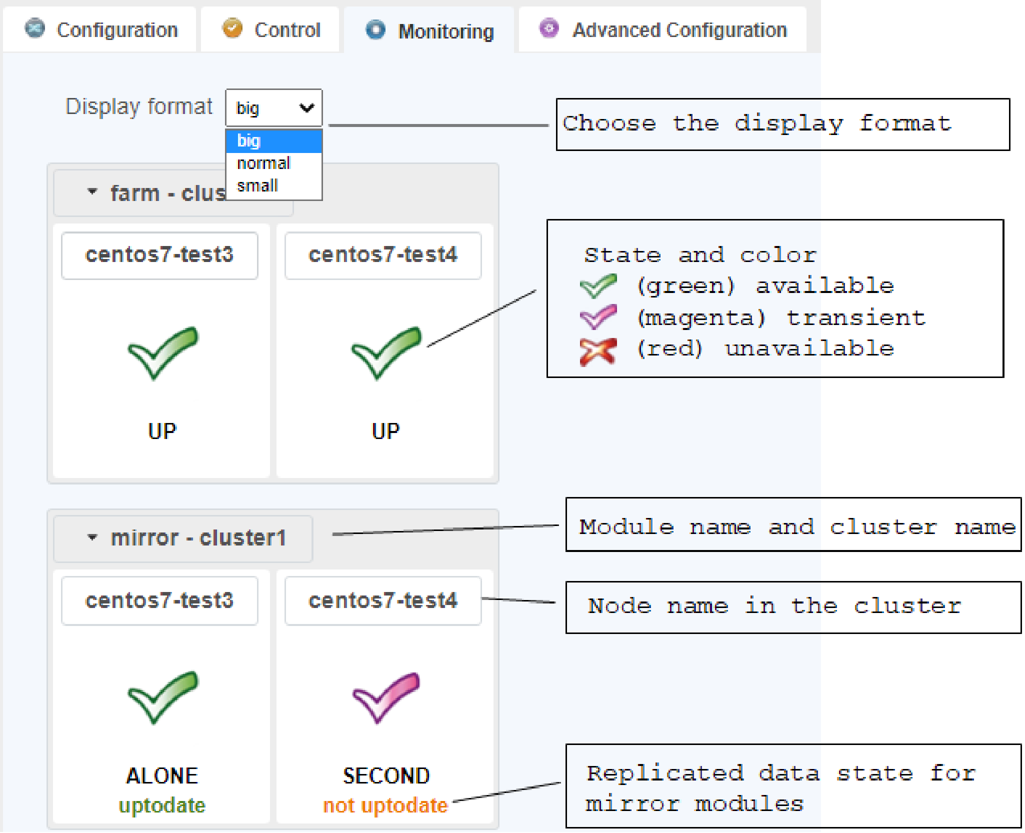
Notes (below `AM` is the module name):

- A dump command creates a directory `dump_year_month_day_hour_mn_sec` on the server side under `SAFEVAR/snapshot/modules/AM`. The `dump` directory contains the module logs (verbose and not verbose) and information on the system state and SafeKit processes at the time of the dump
- A snapshot command creates a dump and gathers under `SAFEVAR/snapshot/modules/AM` the last 3 dumps and last 3 configurations to archive them in a .zip file

3.6 Monitor modules

- In Monitoring tab

  By default, modules that can be monitored are all the modules installed on the nodes from the SafeKit cluster. You can choose the display format for modules according to your needs.



Choose the display format

State and color
✓ (green) available
✓ (magenta) transient
✗ (red) unavailable

Module name and cluster name

Node name in the cluster

Replicated data state for mirror modules

For each module, it displays:

- the current state of the module instances on all cluster nodes are displayed
- You can run a global action on all nodes in the cluster or some local actions only on one node. Click on the ▼ button to open the menu of actions.

For more information on state changes:

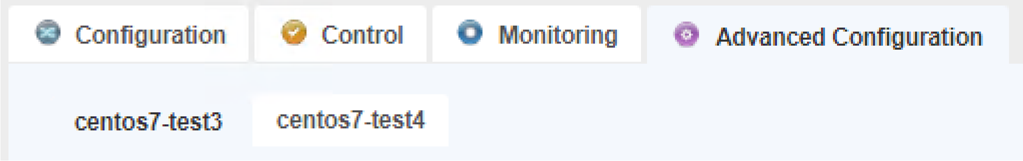
⇨ for a mirror module, see section 5.2 page 96

for a farm module, see section 6.2 page 106

3.7 Manage modules

⇨ In Advanced Configuration tab

There is a tab for each node from the SafeKit cluster. Click on the Node tab for switching server.

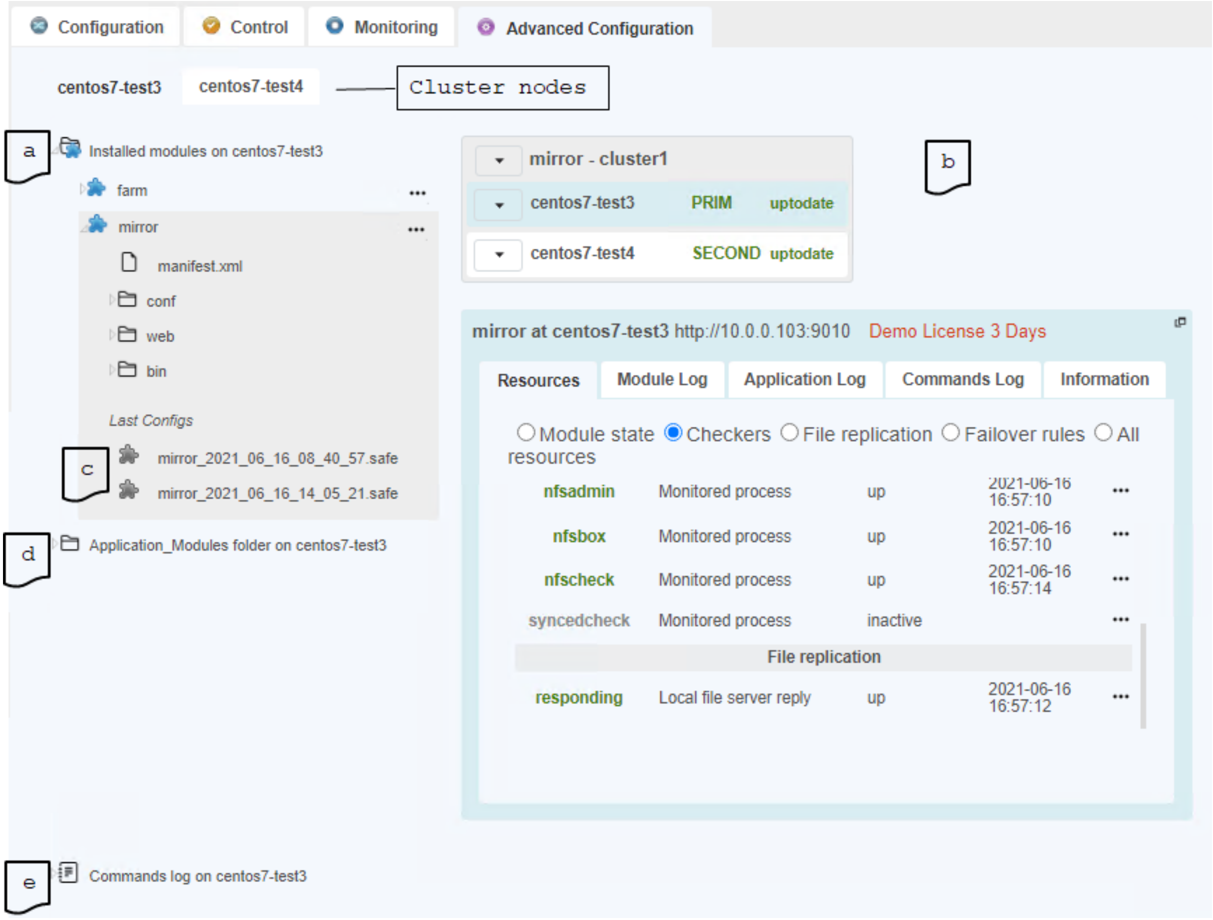


⇨ Node tab

The node tab provides a file and module manager for the server and is divided into 5 areas.

Important

All entries listed on the left panel open a contextual menu on right-click on when clicking on ⋯.


Configuration
Control
Monitoring
Advanced Configuration
centos7-test3
centos7-test4
Cluster nodes
a
Installed modules on centos7-test3
farm
mirror
manifest.xml
conf
web
bin
Last Configs
c
mirror_2021_06_16_08_40_57.safe
mirror_2021_06_16_14_05_21.safe
d
Application_Modules folder on centos7-test3
e
Commands log on centos7-test3
b
mirror - cluster1
centos7-test3 PRIM uptodate
centos7-test4 SECOND uptodate
mirror at centos7-test3 http://10.0.0.103:9010
Demo License 3 Days
Resources
Module Log
Application Log
Commands Log
Information

○ Module state ● Checkers ○ File replication ○ Failover rules ○ All resources

nfsadmin	Monitored process	up	2021-06-16 16:57:10	•••
nfsbox	Monitored process	up	2021-06-16 16:57:10	•••
nfscheck	Monitored process	up	2021-06-16 16:57:14	•••
syncedcheck	Monitored process	inactive		•••
File replication				
responding	Local file server reply	up	2021-06-16 16:57:12	•••

⇨ Installed modules on the selected server

Lists all the modules installed on the selected server (stored into `SAFE/modules` on the server side). Module icon is:

- ✓ 🧩 (blue) when the module is installed, and its configuration is not modified compared to the last applied configuration
- ✓ 🧩 (purple) when the module is installed but at least one of its configuration files has been modified compared to the last applied configuration. In that case, you must apply the new configuration for the changes to take effect
- ✓ 🧩 (grey) when the module is packaged into a unique file .safe

You can browse and edit the module content for advanced configuration (see section 3.7.1 page 58). On right-click, it opens a contextual menu of operations allowed on the installed module (apply the configuration, check configuration …) or on directory/file (common operations: copy, paste ...). Click on a directory to open or close it.

Click on the installed module name for enabling the control panel (b) on the module.

⇨ (b) Control panel of a module installed on the selected server

Allow control of the selected module and detailed view of the module for each node of the cluster. It is like the one provided into the ✔ Control tab.

Note

The Informations tab displays the summary of the active configuration of the module that is the last configuration successfully applied. It may be different from the one into Installed modules tree if configuration has been modified but not applied. In that case, the icon for the module is 🧩 (purple).

⇨ (c) Last Configs of the module installed on the selected server

SafeKit keeps the 3 last successful configuration files for each module (stored in `SAFE/modules/lastconfig` on the server side), packed in a 🧩 .safe file abiding by the `AM_<date>_<time>` naming convention (where AM is the module name). To restore a previous configuration, right click on the 🧩 .safe and select the operation ⊗ Restore the configuration. It opens the Configuration wizard (described in section 3.3.2 page 44) with the content of the saved configuration.

⇨ (d) Application_Modules folder on the selected server

It displays the content of the `SAFE/Application_Modules` directory on the server side.

It is used as a:

- ✓ repository for module templates (in generic, demo and other)
- ✓ backup storage for modules (in backup)
- ✓ workspace for implementing new module templates

To copy installed module and last module configuration files into Application_Modules folder, right-click on the source and select the operation Save to Application_Modules. Right-click on the entry to get more actions.

⇨ (e) Commands Log of the selected server

This area is a nice display of the `safekit` commands that have been executed on the server (see section 10.9 page 175).

3.7.1 Advanced configuration of a module

For advanced configuration, go throw the following steps:

- 3.7.1.1 "Edit configuration files" page 58
- 3.7.1.2 "Apply the configuration" page 59

3.7.1.1 Edit configuration files

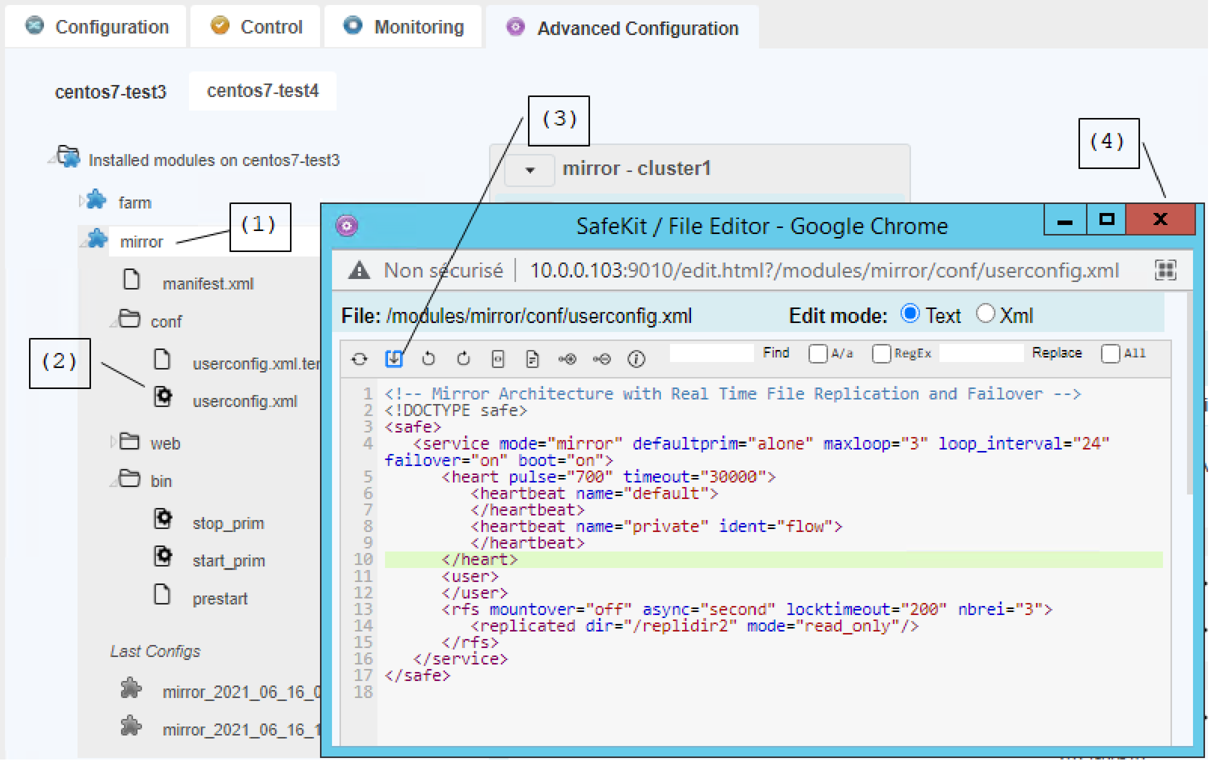
Edit configuration files to:

- set advanced configuration options into `userconfig.xml` (described in section 13 page 235)
- insert your application start/stop into scripts (described in section 14 page 293)

See also examples listed in section 15 page 299.

For this:

- In Advanced Configuration tab
- Node tab
- Installed modules



File: /modules/mirror/conf/userconfig.xml   Edit mode: ● Text ○ Xml

```
<!-- Mirror Architecture with Real Time File Replication and Failover -->
<!DOCTYPE safe>
<safe>
  <service mode="mirror" defaultprim="alone" maxloop="3" loop_interval="24"
failover="on" boot="on">
      <heart pulse="700" timeout="30000">
         <heartbeat name="default">
         </heartbeat>
         <heartbeat name="private" ident="flow">
         </heartbeat>
      </heart>
      <user>
      </user>
      <rfs mountover="off" async="second" locktimeout="200" nbrei="3">
         <replicated dir="/replidir2" mode="read_only"/>
      </rfs>
  </service>
</safe>
```

- Navigate through the module tree (content of the directory `SAFE/modules/AM` on the server side, where `AM` is the module name). Click to open directories
- (2) Click on a file to edit it. `userconfig.xml` is located under `conf`; user scripts are located under `bin`.

  The editor features an XML syntaxic mode for the `userconfig.xml` file. Click on the Text/Xml radio button to toggle between raw text mode and smart XML mode. In XML mode:

  - ✓ clicking on the Insert button activates the insert mode. In this mode, allowed additional tags appear in green, bold font, whereas allowed additional attributes appear in green, italic font. Clicking on an allowed tag or attribute inserts an instance of it at the appropriate place.
  - ✓ clicking on the Erase button activates the erase mode. In this mode, clicking on a tag or attributes removes it (and its children) from the edited document. When the mouse is over such an element, the part of the document that would be removed on mouse click is highlighted in red, stricken text.

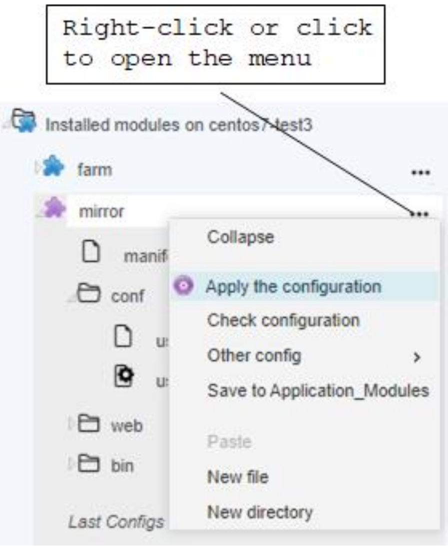
- (3) Save the modifications to the server
- (4) Close the editor window

For a first test of SafeKit:

- apply this procedure to mirror or farm module
- open `userconfig.xml` file to see the module configuration
- open start and stop scripts

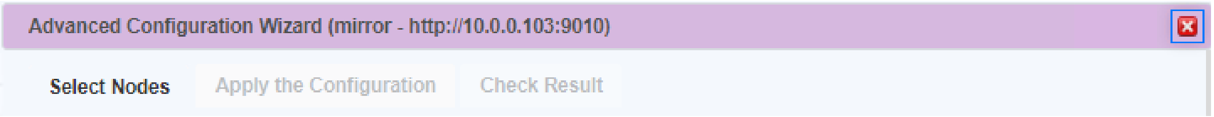
3.7.1.2 Apply the configuration

- In Advanced Configuration tab
- Node tab
- Installed modules
- Right-click or click ··· on the module entry to open the menu
- Choose Apply the configuration that opens the Advanced Configuration wizard



3.7.2 Advanced configuration wizard

The wizard walks you through the process of advanced configuration of a module. It allows applying the configuration of a module that can be modified by directly editing configuration files (see section 3.7.1.1 page 58).


Advanced Configuration Wizard (mirror - http://10.0.0.103:9010)
Select Nodes
Apply the Configuration
Check Result

⇨ 3.7.2.1 "Select nodes" page 60

Use this tab to select the nodes on which to apply the configuration. It may be useful to apply the configuration on only one node, for testing the new configuration on that node before applying it to other nodes in the cluster.

⇨ 3.7.2.2 "Apply the configuration on selected nodes" page 61

The configuration is applied only on the previously selected nodes.

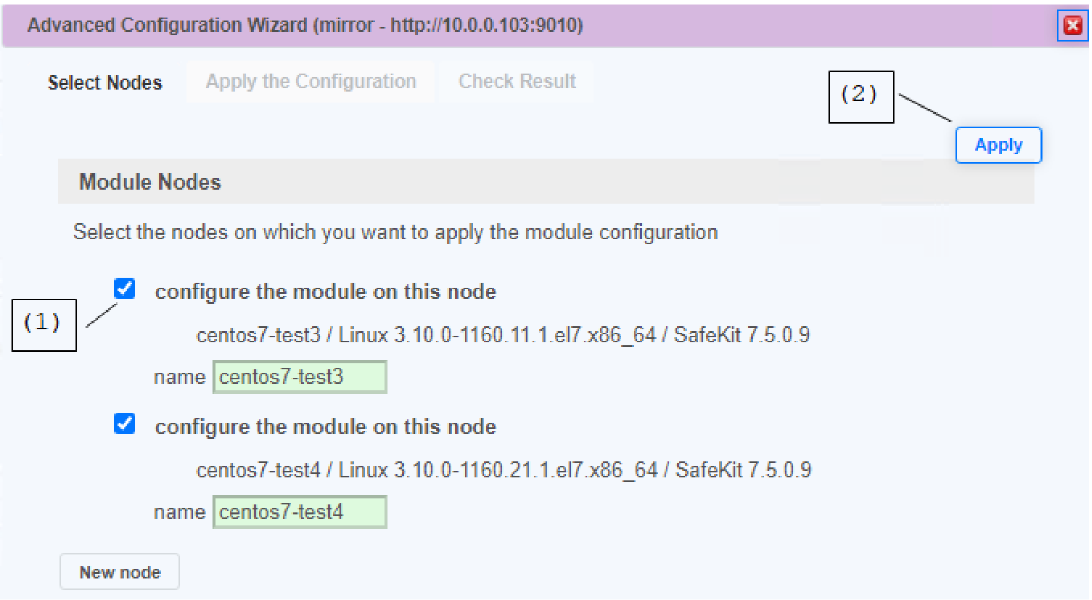
⇨ 3.7.2.3 "Check the configuration result" page 61

This tab is enabled when the configuration has been applied in the previous step.

3.7.2.1 Select nodes

⇨ In Advanced Configuration wizard

⇨ Select Nodes tab



Select the nodes on which you want to apply the module configuration

☑ configure the module on this node

centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9

name centos7-test3

☑ configure the module on this node

centos7-test4 / Linux 3.10.0-1160.21.1.el7.x86_64 / SafeKit 7.5.0.9

name centos7-test4

By default, all the module cluster nodes are selected.

⇨ Check the box to apply the configuration on the node; uncheck for not applying it.

- If necessary, add a new node in the same way as described in the configuration wizard (as described in section 3.3.2.1 page 44).
- (2) Click on the Apply button when the form has been modified and go to the next step

3.7.2.2 Apply the configuration on selected nodes

- In Advanced Configuration wizard
- Apply the configuration tab

This tab is the same as the one into the configuration wizard (described in section 3.3.2.3 page 48), but with the following major differences:

- ✓ Only the nodes selected in the previous step are displayed
- ✓ When clicking on the Apply button, there is no check that the module is stopped on all nodes. That implies that the configuration may be applied while the module is running. In that case, there is an attempt to make a dynamic reconfiguration. This one is successful only if:
    - The module is in the `ALONE` (green) or `WAIT` (red) states
    - You have modified into the `userconfig.xml` file, only parameters that can be dynamically changed (see section 13 page 235)

Important

If you do not want to run dynamic configuration, stop the module on all nodes before clicking on the Apply button.

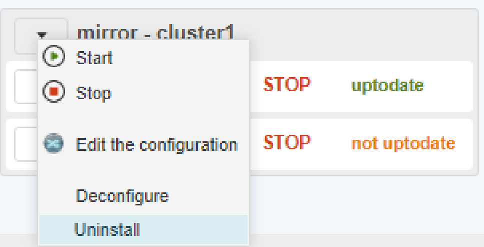
3.7.2.3 Check the configuration result

- In Advanced Configuration wizard
- Check Result tab

This tab is the same as the one into the configuration wizard (described in section 3.3.2.4 page 49).

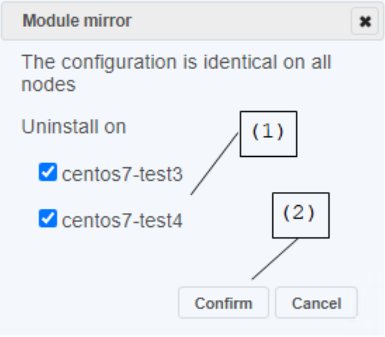
3.7.3 Uninstall a module

- In Configuration or Advanced Configuration tabs
- Click on the ▾ button of the module to open the menu on the module and select Uninstall



It opens a dialog for selecting the nodes on which the module will be uninstalled.

- (1) Check the box for uninstalling the module on the node; uncheck for not uninstalling
- (2) Click on the Confirm button to run uninstall on the selected nodes



Module mirror

The configuration is identical on all nodes

Uninstall on

☑ centos7-test3

☑ centos7-test4

Note:

- before uninstalling, close all editors, file explorers, shells or cmd under `SAFE/modules/AM` and `SAFEVAR/modules/AM` (where AM is the module name). Otherwise, there is a risk that uninstalling the module fails
- after uninstalling, the uninstalled module is stored under the `SAFE/Application_Modules/backup` directory on the server side

The command line sequence equivalent for uninstalling is (replace below AM by the module name):

1. Log as administrator/root and open a command shell window on one node

   For instance, log-in `node1`

2. Run `safekit deconfig -m AM`

   To unconfigure the module `AM` locally

3. Run `safekit module package -m AM SAFE/Application_Modules/backup/AM.safe`

   To save the module in `SAFE/Application_Modules/backup/AM.safe`

4. Run `safekit module uninstall -m AM`

   To uninstall the module locally

Repeat all these commands on the other node.

3.7.4 Configure a module stored into Application_Modules

You may need to configure a module from a template stored into Application_Modules such as a template (in demo folder), a backup (in backup/) or a saved module. For this:

- In Advanced Configuration tab
- Node tab
- Application_Modules folder
- Right-click on .safe, then Edit the configuration. It opens the Configuration wizard (described in section 3.3.2 page 44). This wizard offers only quick configuration for the module. If you need advanced configuration, Unpack the .safe and edit the files into the extracted module tree.
- Right-click on the unpacked module entry, then Apply the configuration. It opens the advanced configuration wizard (described in section 3.7.1 page 58).

You can store an external module into the Application_Modules area for configuring it later:

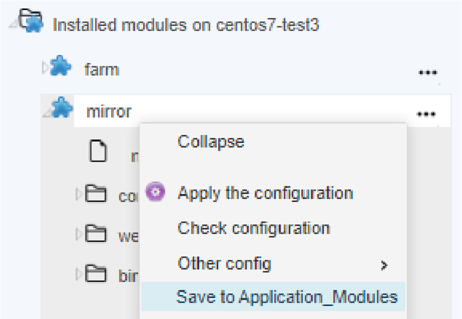
- In Advanced Configuration tab
- Node tab
- Right-click on Application_Modules folder, then Load .safe from your workstation storage

3.8 Create a new module template (.safe) for deployments

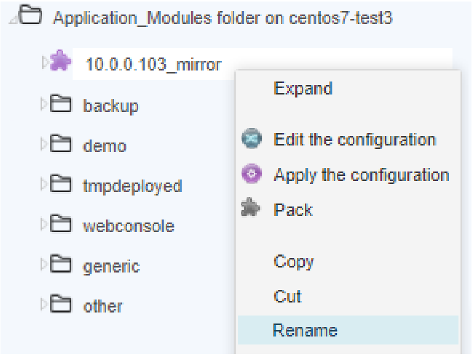
Once your module is built and validated, you may want to reuse it on a new customer site. For this, create first the module template on your test site, and then deploy it on the new site.

3.8.1 Create a new module template

- In Advanced Configuration tab
- Node tab
- Installed modules
- Right-click on the module you want to publish and choose Save to Application_Modules



- Go to Application_Modules folder
- Right-click on the copy of the module and Rename it with the name of the new template (appli for instance)

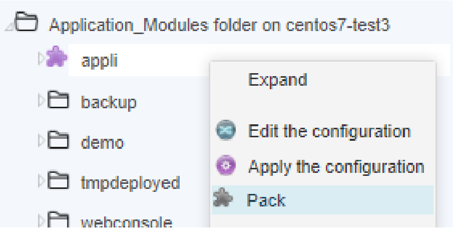


You now must modify the configuration files of the module so that it can be used as a template:

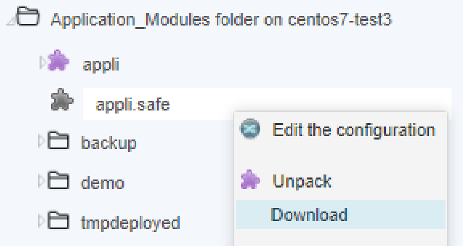
- Browse appli folder for editing files
- Edit `conf/userconfig.xml` to remove values specific to your cluster. Click on the file to open the editor. It opens the file for editing as text.
- Edit other files if they also contain parameters specific to your installation

Optionally, edit `web/index.html` to change the html page displayed in Enter Parameters tab of Configuration wizard. When `index.html` is not present (in previous versions modules for instance), the web console instead proposes to edit the `userconfig.xml` file.

- Right-click on the module, then Pack. It generates the .safe file in the folder.



- Save the template on your workstation. For this, right-click on appli.safe, then Download. The Web console relies on the web browser download settings for saving the file on your workstation



3.8.2 Deploy a new module template

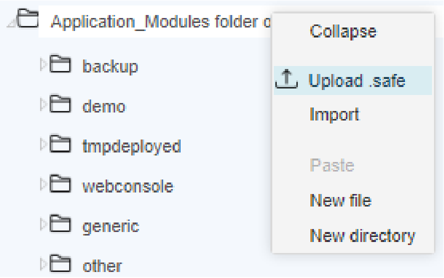
After a new SafeKit installation on new servers, you can make your module template available from Configuration tab.

- Manually copy the file appli.safe on the new SafeKit server in `SAFE/Application_Modules/demo` (create it if necessary)

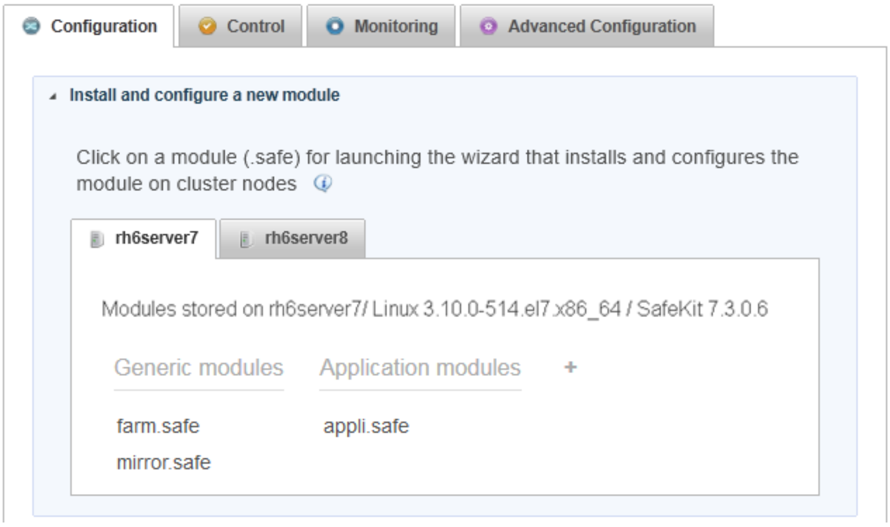
Or

- Connect the web console to the new SafeKit server
- In Advanced Configuration tab
- Node tab
- Right-click on Application_Modules folder

  Choose Load .safe from your workstation storage. Then move appli.safe into demo



At the next web console connection to the new server (or after refreshing the web page), appli.safe is visible in Configuration tab.



Install and configure a new module

Click on a module (.safe) for launching the wizard that installs and configures the module on cluster nodes

Modules stored on rh6server7/ Linux 3.10.0-514.el7.x86_64 / SafeKit 7.3.0.6

Generic modules	Application modules	+
farm.safe	appli.safe	
mirror.safe		

The module template appli.safe can be installed and configured in the same way as other module templates (see section 3.3 page 41).

3.9 Secure access to the web console

SafeKit offers different security policies for the web console that are implemented by modifying the SafeKit web service configuration. These configurations also offer role management:

Admin role ● ● ● ●	This role grants all administrative rights by allowing access to the tabs: ● Configuration, ● Control, ● Monitoring and ● Advanced Configuration
Control role ● ●	This role grants control and monitoring rights by allowing access to the tabs: ● Control et ● Monitoring
Monitor role ●	This role only grants monitoring rights by allowing access to the tab: ● Monitoring

SafeKit provides different setups for the web service to enhance the security of the SafeKit web console. The predefined setups are listed below from least secure to most secure:

- HTTP. Same role for all users without authentication

  This solution can only be implemented only in HTTP and is not compatible with user authentication methods.

- HTTP/HTTPS with user authentication based on Apache files and optional role management

  It relies on Apache files to store username/password for authenticating users and, optionally, to store the associated role for restricting their access. To connect to the console, the user must enter the username and password as configured with the Apache mechanisms.

Since SafeKit 7.5, this is the default active configuration, applied for HTTP and initialized with a single `admin` user with the Admin role. The default setup can be extended to add users or to switch to HTTPS.

- HTTP/HTTPS with user authentication based on LDAP/AD authentication. Optional role management

  It relies on LDAP/AD authentication server to authenticate users and, optionally, restricts their access based on roles. To connect to the console, the user must enter the username and password as configured into the LDAP/AD server. It supports HTTP or HTTPS.

- HTTPS with client certificate authentication and role management

  It relies on client certificates to authenticate users and assign their role. To connect to the console, the user must import the appropriate client certificate into its browser. It requires HTTPS configuration.

To implement them, refer to the section 11 page 177.

3.10 The clusters inventory of the web console

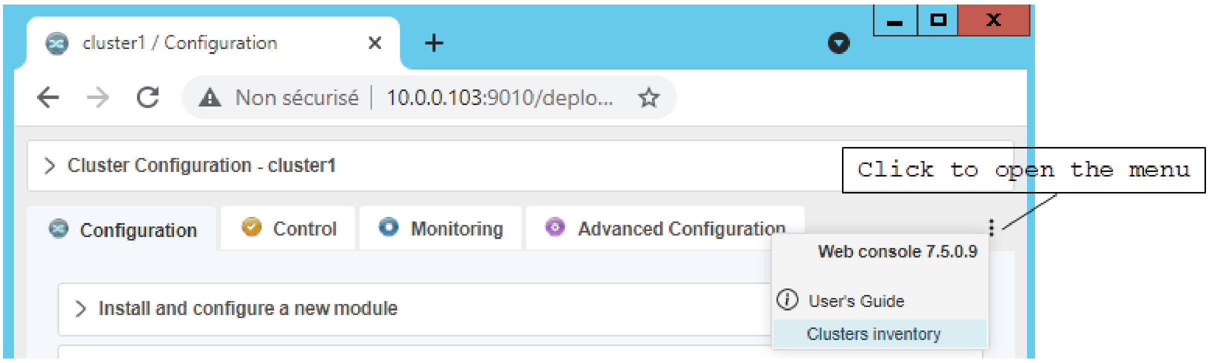
Each entry in the inventory corresponds to a SafeKit cluster and points to one of the nodes in the cluster. This is the main connection used by the web console to get this cluster configuration and administration.

Important

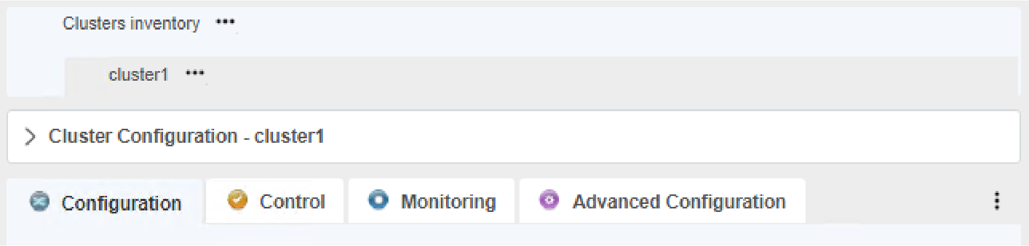
Clusters inventory is stored into the cache of the web browser. Therefore, it must be reassigned after the cache clean or when using another browser.

3.10.1 Define the clusters inventory of the web console

To display the inventory, click on ⋮ to open the menu, then select Clusters inventory. Apply the same procedure to hide the inventory.

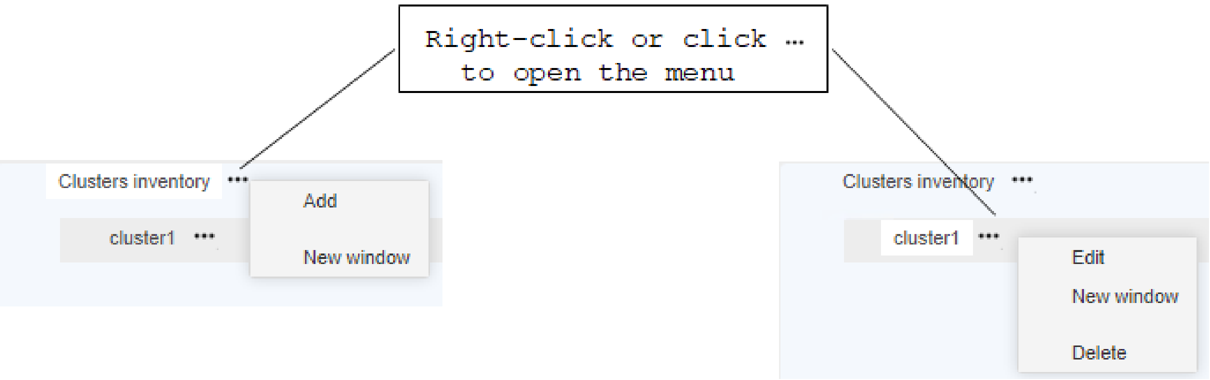


It displays the clusters inventory panel at the top of the page:

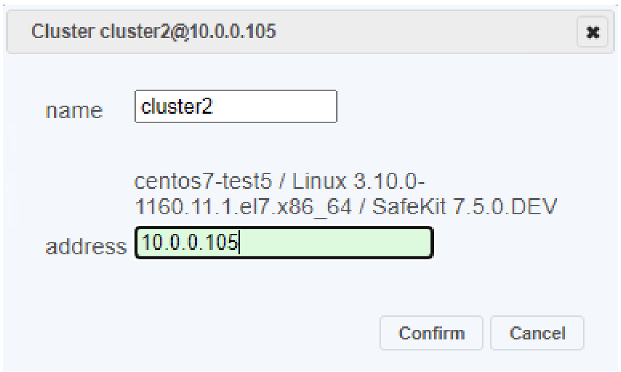


On the first connection with the web console, the connection server is automatically added into the inventory with the name `cluster1`.

- In Clusters inventory
- Right-click or click on ··· of the entry to open the menu and edit, delete or add an entry to the inventory


Right-click or click … to open the menu

You can edit `cluster1` or add a new entry `cluster2`:


Cluster cluster2@10.0.0.105

name: cluster2

centos7-test5 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.DEV

address: 10.0.0.105

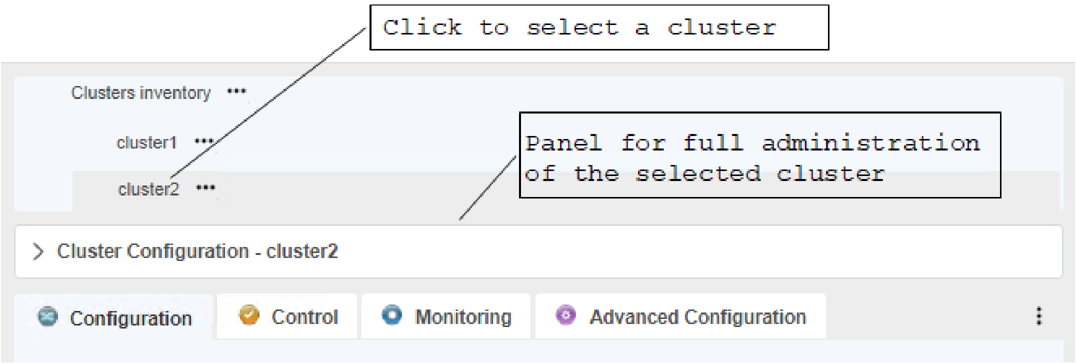
- Set the name of the cluster. This name is displayed into the web console for identifying the cluster
- Check the server identity
- Click on Confirm to add/change the cluster to the inventory

Note

The clusters inventory can also been set with a query string in the URL. For instance: `http://172.24.199.107:9010/deploy.html?inventory=cluster1@172.24.199.107,cluster2@172.24.199.105` open the console with the inventory set in parameter.

3.10.2 Administer one cluster of the inventory with the web console

⇨ In Clusters inventory panel

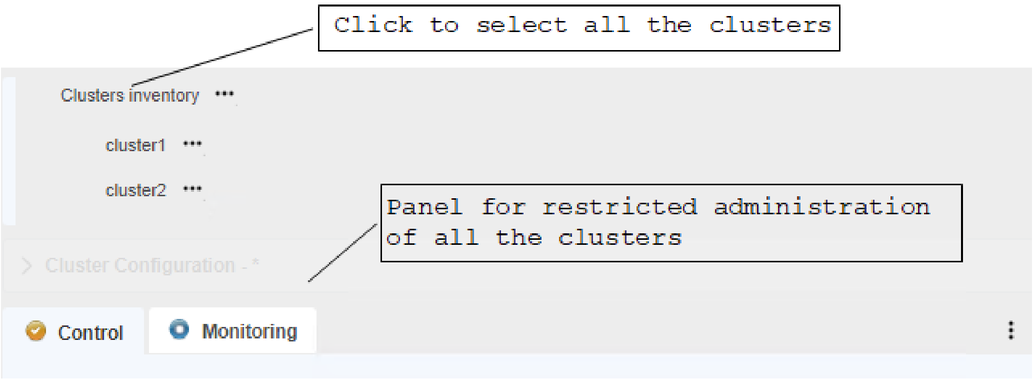

Click to select a cluster
Clusters inventory •••
cluster1 •••
cluster2 •••
Panel for full administration of the selected cluster
Cluster Configuration - cluster2
Configuration
Control
Monitoring
Advanced Configuration

⇨ (1) Click on the name of the cluster you want to administer

⇨ (2) The administration panel displays the full tools for the selected cluster

3.10.3 Administer all the clusters of the inventory with the web console

⇨ In Clusters inventory panel


Click to select all the clusters
Clusters inventory •••
cluster1 •••
cluster2 •••
Panel for restricted administration of all the clusters
Cluster Configuration - *
Control
Monitoring

⇨ (1) Click on the Clusters inventory entry

⇨ (2) The administration panel displays a restricted set of tools, but for all the clusters

Since SafeKit 7.5, this global administration of modules from all clusters is incompatible with the configuration of user authentication based on file or LDAP/AD server. This means that it is incompatible with the default configuration of the SafeKit web service. If you need this feature, change the default configuration to the unsecure one or the secured one based on HTTPS and client certificates. Refer to section 11 page 177.

4. Tests

4.1 Installation and tests after boot

4.1.1 Test package installation

Package installation:

- ➾ `safekit -p` returns:

  Windows:
  ```
  "SAFE=C:\safekit"
  "SAFEVAR=C:\safekit\var"
  ```
  Linux :
  ```
  "SAFE=/opt/safekit"
  "SAFEVAR=/var/safekit"
  ```

- ➾ **`SAFE`** – Installation directory of SafeKit: **`C:\safekit`** on Windows if `%SYSTEMDRIVE%`=C:, **`/opt/safekit`** on Linux
- ➾ Editing **`userconfig.xml`** of a mirror(/farm) module and its scripts **`start_prim/start_both`**, **`stop_prim/stop_both`** is made with the web console/⚙ Advanced Configuration/ (see section 3.7.1 page 58) or inside `SAFE/modules/AM` (where AM is the module name)
- ➾ **Output messages of application scripts** are in the web console/✓ Control /Select the node /Application Log tab/ or in `SAFEVAR/modules/AM/userlog.ulog` (where AM is the module name). Check if there are errors during start or stop of the application. Be careful, sometimes the userlog is disabled because it is too large with <user logging="none"> in `userconfig.xml` of the module
- ➾ For more information, see section 10.1 page 153

4.1.2 Test license and version

- `safekit level` returns

  Host : <hostname>
  OS : <OS version>
  SafeKit : <SafeKit version>
  License : Demo (No license)| Invalid Product | Invalid Host | … Expiration… | <license id> for <hostname>…
  or License : Expired license

- "Demo (No license)" means no `SAFE/conf/license.txt` file: the product stops every 3 days
- "Invalid Product" means an expired license in `SAFE/conf/license.txt`
- "Invalid Host" means no valid hostname in `SAFE/conf/license.txt`
- " …Expiration…" means a temporary key
- "<license id> for <hostname>" means a permanent license
- http://www.evidian.com/safekit/requestevalkey.php to get a temporary key of one month for any OS or any hostname
- https://support.evidian.com to get a permanent key based on the hostname and OS

4.1.3 Test SafeKit services and processes running after boot

See also section 9.2 page 142.

Test `safeadmin` service:

- The `safeadmin` process must appear in the list of running processes
- Without this process, no `safekit` command works and they all return:

  "Waiting for `safeadmin` ...."
  "Error: `safeadmin` administrator daemon not running"
- On Windows, `safeadmin` is a service and can be started in the Services interface of Windows
- on LINUX, `safeadmin` is started by *service `safeadmin` start* on Linux

Test `safewebserver` service:

- `safekit boot webstatus` displays start-up or not of `safewebserver` service at boot ("on" or "off", "on" by default)
- httpd processes must be in the list of running processes if boot "on"
- without these processes, the web console is not able to connect to servers as well <module> checkers (`userconfig.xml`) and distributed command line interface
- to start/stop the `safewebserver` service, run: `safekit webserver start|stop`

Test `safeagent` service:

- `safekit boot snmpstatus` displays start-up or not of `safeagent` service at boot ("on" or "off", "off" by default)
- `safeagent` process must be in the list of running processes if boot "on"
- to start/stop the `safeagent` service, run: `safekit safeagent start|stop`

Test modules:

- `safekit boot status` displays start-up ("on") or not ("off") of modules at boot
- `safekit state` displays state of all configured modules: `STOP` (mirror or farm), `WAIT` (mirror or farm), `ALONE` (mirror), `PRIM` (mirror), `SECOND` (mirror), `UP` (farm)
- check processes of a module: see section 10.2 page 155
- `safekit module listid` displays name of installed modules with their ids: id of a module must be the same on all servers
- go to `SAFE/modules/AM/conf` (replace AM by the module name); `userconfig.xml` file gives the module type, mirror, or farm: <service mode="mirror"> or <service mode="farm">

4.1.4 Test start of SafeKit web console

- connect a web browser to http://<server IP>:9010
- the web console home page is displayed

4.2 Tests of a mirror module

4.2.1 Test start of a mirror module on 2 servers ✗ STOP (red)

- message in the logs of both servers (web console/✔ Control/Select the node/Module Log tab/ or the command `safekit logview -m AM` where AM is the module name)

  "Action start called by web@<IP>/SYSTEM/root"

- the module goes to the stable state ✓ `PRIM` (green) and ✓ `SECOND` (green) on both servers with in the first log

  "Remote state `SECOND` green"
  "Local state `PRIM` green"

- and in the other log

  "Local state `SECOND` green"
  "Remote state `PRIM` green"

- application is started in the `start_prim` script of the module on the `PRIM` server with message in the log

  "Script `start_prim`"

4.2.2 Test stop of a mirror module on the server ✓ PRIM (green)

- message in the log of the stopped server (web console/✔ Control/Select the node/Module Log tab/ or the command `safekit logview -m AM` where AM is the module name)

  "Action stop called by web@<IP>/SYSTEM/root"

- the stopped server runs the stop_prim script of the module which stops the application on the server with message in the log:

  "Script stop_prim"

- the module becomes ✗ `STOP` (red) with messages in the log:

  "End of stop"

  "Local state `STOP` red"

- the module becomes ✓ `ALONE` (green) on the other server with the message in the log:

  "Reason of failover: remote stop"

- the application is started with the `start_prim` script on the `ALONE` server with the message in the log:

  "Script `start_prim`"

4.2.3 Test start of a mirror module on the server ✗ STOP (red)

- message in the log of the started module (web console/✅ Control/Select the node/Module Log tab/ or the command `safekit logview -m AM` where AM is the module name)

  "Action start called by web@<IP>/SYSTEM/root"

- the red `STOP` module becomes ✓ `SECOND` (green)
- the module ✓ `ALONE` (green) on the other server becomes ✓ `PRIM` (green) and continues to execute the application

4.2.4 Test restart of a mirror module on the server ✓ PRIM (green)

- message in the log of the server where the restart command is passed (web console/✅ Control/Select the node/Module Log tab/ or the command `safekit logview -m AM` where AM is the module name)

  "Action restart called by web@<IP>/SYSTEM/root"

- the `PRIM` module becomes ✓ `PRIM` (magenta) and then becomes ✓ `PRIM` (green)
- the scripts of the module `stop_prim/start_prim` are executed on the `PRIM` module and restarts locally the application on the server with messages in the log:

  "Script `stop_prim`"
  "Script `start_prim`"

- the other module on the other server stays ✓ `SECOND` (green)

4.2.5 Test swap of a mirror module from one server to the other

- message in the log of the server where the `swap` command is passed (web console/ ✅ Control/Select the node/Module Log tab/ or the command `safekit logview -m AM` where AM is the module name)

  "Action swap called by web@<IP>/SYSTEM/root"

  "Transition SWAP from SYSTEM"

  "Begin of Swap"

- And in the log of the other server, only:

  "Begin of Swap"

- reversing the roles of `PRIM` and `SECOND` between both servers
- the stop_prim script is first executed on the former `PRIM` within its log:

  "Script stop_prim"

- then the `start_prim` script is executed on the new `PRIM` server with in its log:

  "Script `start_prim`"

- at the end of `swap`, module ✓ `PRIM` (green) and module ✓ `SECOND` (green) are reversed on both servers and the application is on the new `PRIM` server

4.2.6 Test virtual IP address of a mirror module

Mirror module in the state ✓ `PRIM` (green) on server node1 and ✓ `SECOND` (green) on server node2.

`userconfig.xml`:

```
<vip>
 <interface_list>
 <interface arpreroute="on">
 <real_interface>
   <virtual_addr addr="ipvirt"
             where="one_side_alias"/>
  </real_interface>
  </interface>
 </interface_list>
</vip>
```

1. On an external workstation (or server) in the same LAN, ping both physical IP addresses + virtual IP address:

   ```
   ping node1_ip_address
   ping node2_ip_address
   ping ipvirt
   arp –a
   ```

2. `safekit swap –v AM` on the primary server (where AM is the module name)

3. On the external workstation (or server),

   ```
   ping node1_ip_address
   ping node2_ip_address
   ping ipvirt
   arp –a
   ```

   Note: redo the ping to virtip before looking at the ARP table because the entry may be marked obsolete and refreshes only after ping

1. On server node1, `ipconfig` or `ifconfig` (or `ip addr show`) returns ipvirt as an alias on the network interface.

   On the external workstation (or server), the 3 pings respond

   On the external workstation (or server) in the same LAN, virtip is mapped to the same MAC address as node1_ip_address

   ```
   arp –a
   node1_ip_address       00-0c-29-0a-5c-fc
   node2_ip_address       00-0c-29-26-44-93
   ipvirt      00-0c-29-0a-5c-fc
   ```

2. After the `swap`, ✓ `SECOND` (green) on node1 server and ✓ `PRIM` (green) on node2 server

   In the log of new primary, message:

   ```
   "Virtual IP <ipvirt of mirror> set"
   ```

3. On node2, `ipconfig` or `ifconfig` (or `ip addr show`) returns ipvirt as an alias on the network interface

   On the external workstation (or server), the 3 pings respond

   On the external workstation (or server), virtip is mapped to the same MAC address as node2_ip_address

   ```
   arp –a
   node1_ip_address       00-0c-29-0a-5c-fc
   node2_ip_address       00-0c-29-26-44-93
   ipvirt       00-0c-29-26-44-93
   ```

4.2.7 Test file replication of a mirror module

Mirror module in the state ✓ PRIM (green) on node1 server and ✓ SECOND (green) on node2 server.

```
userconfig.xml:
<rfs>
<replicated dir="C:\replicated"
mode="read_only" />
                        (or "/replicated"
on Linux)
</rfs>
```

1. On the server ✓ PRIM (green), go to /replicated and create a file file1.txt
2. On the server ✓ SECOND (green), go to /replicated and try to delete file1.txt
3. Stop the server ✓ PRIM (green) and wait for ✗ STOP (red). Then go to the other server which is ✓ ALONE (green) and create a new file file2.txt
4. Restart the server ✗ STOP (red) and wait for ✓ SECOND (green).

1. file1.txt has been replicated on ✓ SECOND (green) under /replicated
2. Failure because the /replicated directory is read-only on the server ✓ SECOND (green)
3. file2.txt is not replicated in /replicated of the server ✗ STOP (red)
4. file2.txt is reintegrated on the restarted server. During the phase of reintegration, the server is ✓ SECOND (magenta)

   In the log of reintegrated server, message

   "Updating directory tree from /replicated"

   And at the end of /replicated reintegration, if at least 1 file with modified data has been reintegrated from primary server to secondary server, message

   "Copied <reintegration statistics>"

   "Reintegration ended (synchronize)"

   This message gives statistics for the reintegrated directory: reintegrated size, number of files, time, and throughput on the network in KB/sec.

   Note: reintegrate a file larger than 100 MB to have reliable statistics

   At the end of reintegration, the server is ✓ SECOND (green)

4.2.8 Test mirror module shutdown on the server ✓ PRIM (green)

- on Windows, check that the special procedure to stop modules at shutdown has been applied.
- make a shutdown of ✓ `PRIM` (green) server
- check in the log of server ✓ `SECOND` (green), message

  "Reason of failover: remote stop"
- the server ✓ `SECOND` (green) becomes ✓ `ALONE` (green); application in the `start_prim` script of the module is restarted on the `ALONE` server with the message in the log

  "Script `start_prim`"
- on timeout in the SafeKit console, the old server ✓ `PRIM` (green) becomes grey
- after reboot of the stopped server, check that the OS shutdown has really called a shutdown of the module

  "Action shutdown called by SYSTEM"
- Check that the application `stop_prim` script has been executed with the message

  "Script stop_prim"
- And check that the module has been completely stopped before shutting down the server with the last message

  "End of stop"
- after reboot of stopped server, if the module is started automatically at boot (`safekit boot status`), message in the log

  "Action start called at boot time"
- after a start of the module on the stopped server, the module becomes ✓ `SECOND` (green) on this server and ✓ `PRIM` (green) on the other server

4.2.9 Test mirror module power-off on the server ✓ PRIM (green)

userconfig.xml:

```
<heart>
 <heartbeat name="default" />
 <heartbeat ident="flow" />
</heart>
```

Note: If you want to make a test with double simultaneous electrical fault on both servers, check that <rfs async="none"> is set in userconfig.xml. For more information, see section 1.3.6 page 18

- in the log of the server ✓ SECOND (green), message for all heartbeats configured in userconfig.xml

  "Resource heartbeat.default set to down by heart"
  "Resource heartbeat.flow set to down by heart"
  "Remote state UNKNOWN grey"
  "Reason of failover: no heartbeat"

- messages appear within 30 seconds after the power-off (if no specified timeout configured for <heart> in userconfig.xml)
- the server ✓ SECOND (green) becomes ✓ ALONE (green); the application in the start_prim script of the module is restarted on the ALONE server with the message in its log

  "Script start_prim"

- on timeout in the SafeKit console, the former server ✓ PRIM (green) becomes grey
- after reboot of stopped server, if the module is started automatically at boot (safekit boot status), message in the log

  "Action start called at boot time"

- after reboot, message in the log:

  "Previous halt unexpected"

- after restart of the module on the stopped server, the module becomes ✓ SECOND (green) on this server and ✓ PRIM (green) on the other server

4.2.10 Test split brain with a mirror module

Split brain occurs in situation of network isolation between two SafeKit servers. Each server becomes primary ALONE and runs the application. At return of split brain, a sacrifice must be made by shutting down the application on one of the two servers.

Mirror module in the state ✓ PRIM (green) and ✓ SECOND (green)

userconfig.xml:

```
<heart>
 <heartbeat name=”default” />
 <heartbeat name=”repli” ident="flow"
/>
</heart>
```

+

on Windows to manage the IP conflict on the virtual IP address virtip

```
<vip>
 <interface_list>
  <interface check="on"
arpreroute="on">
   <real_interface>
    <virtual_addr addr="192.168.1.10"
       where="one_side_alias"/>
   </real_interface>
  </interface>
 </interface_list>
</vip>
```

To obtain the split brain, check that there are no checkers in userconfig.xml that can detect the network isolation: no <interface check="on">, no <ping> checker

1. disconnect all heartbeat networks at the same time (network default and repli)
2. reconnect networks

- after network isolation of both servers, all heartbeats are lost. In the logs of both servers,

  "Resource heartbeat.default set to down by heart"
  "Resource heartbeat.flow set to down by heart"
  "Remote state UNKNOWN grey"
  "Local state ALONE green"

- split brain case: both servers are ✓ ALONE (green) and run the application started in start_prim
- when reconnecting heartbeat networks, sacrifice of one ALONE server: the former SECOND server
- log of the former PRIM not sacrificed:

  "Remote state ALONE green"
  "Split brain recovery: staying alone"

- log of the former SECOND sacrificed:

  "Remote state ALONE green"
  "Split brain recovery: exiting alone"
  "Script stop_prim"

  The server performs a stopstart: stop of the application with stop_prim then reintegration of replicated files from the other server

- come back to the stable state ✓ PRIM (green) and ✓ SECOND (green) on both servers as it was before split brain

Note: situation of split brain in a mirror module with file replication is not good. Indeed, the sacrifice of the former secondary server causes file reintegration of this server from the primary one and the loss of data stored on the secondary during the split-brain situation.

For this reason, 2 heartbeats on two physically separate networks are recommended. Typically, a cable between the two servers will allow (1) to avoid split brain with an additional heartbeat network and (2) set the replication flow on a separate network

4.2.11 Continue your mirror module tests with checkers

Go to section 4.4 page 86 for tests of checkers.

4.3 Tests of a farm module

4.3.1 Test start of a farm module on all servers ✗ STOP (red)

- message in the logs of all servers (web console / logview command)

  "Action start called by web@<IP>/SYSTEM/root"
- the module goes to ✓ `UP` (green) on all servers
- the application is started in the `start_both` script of the module on all servers with the message in the log

  "Script `start_both`"

4.3.2 Test stop of a farm module on one server ✓ UP (green)

- message in the log of the stopped server (web console / logview command)

  "Action stop called by web@<IP>/SYSTEM/root"
- the stopped module runs the `stop_both` script which stops the application on the server and with message in the log

  "Script `stop_both`"
- the stopped module becomes ✗ `STOP` (red) with messages in the log:

  "End of stop"

  "Local state `STOP` red"
- the other servers stay ✓ `UP` (green) and continue to run the application
- restart the module ✗ `STOP` (red) with the start command

4.3.3 Test restart of a farm module on one server ✓ UP (green)

- message in the log of the module where the restart command is passed (web console / logview command)

  "Action restart called by web@<IP>/SYSTEM/root"
- the restarted module becomes ✓ `UP` (magenta) then becomes ✓ `UP` (green)
- the module scripts `stop_both`/`start_both` are executed on the server and restart locally the application with messages in the log

  "Script `stop_both`"
  "Script `start_both`"

4.3.4 Test virtual IP address of a farm module

4.3.4.1 Configuration with vmac_invisible

Farm module in the ✓ `UP` (green) state on 2 servers node1 and node2

`userconfig.xml` with load balancing on the `safewebserver` service (TCP port 9010):

```
<farm>
<lan name=”default” />
</farm>

<vip>
 <interface_list>
  <interface>
  <virtual_interface
type="vmac_invisible" >
    <virtual_addr
addr="virtip" where="alias"/>
  </virtual_interface>
  </interface>
 </interface_list>

<loadbalancing_list>
<group name="FarmProto">
  <rule port="9010"
proto="tcp" filter="on_port"/>
</group>
</loadbalancing_list>
</vip>
```

On a remote workstation (or server) in the same LAN, ping of the 2 physical IP addresses + virtual IP + arp –a

- In the log of all servers:

  "Vitual IP <virtip of farm> set"

- On the 2 servers, `ipconfig` or `ifconfig` (or `ip addr show`) returns virtip as an alias on the network interface

- On a remote workstation (or server), the pings respond. And virtip is mapped with the invisible virtual MAC address:

  ```
  ping node1_ip_address; ping node2_ip_address ; ping
  virtip; arp –a
  node1_ip_address        00-0c-29-0a-5c-fc
  node2_ip_address        00-0c-29-26-44-93
  virtip       5a-fe-c0-a8-38-14
  ```

- Note: by default, the virtual MAC address is a unicast Ethernet address built with 5A:FE (SAFE) and the virtual IP address in hexadecimal

4.3.4.2 Configuration with vmac_directed

Farm module in the ✓ `UP` (green) state on 2 servers node1 and node2

`userconfig.xml` with load balancing on the `safewebserver` service (TCP port 9010):

```
<farm>
<lan name=”default” />
</farm>

<vip>
 <interface_list>
  <interface arpreroute=”on”>
  <virtual_interface
type="vmac_directed" >
    <virtual_addr
addr="virtip" where="alias"/>
  </virtual_interface>
  </interface>
 </interface_list>

<loadbalancing_list>
<group name="FarmProto">
  <rule port="9010"
proto="tcp" filter="on_port"/>
</group>
</loadbalancing_list>
</vip>
```

On a remote workstation (or server) in the same LAN, ping of the 2 physical IP addresses + virtual IP + arp –a

- In the log of all servers:

  "Vitual IP <virtip of farm> set"

- On the 2 servers, `ipconfig` or `ifconfig` (or `ip addr show`) returns virtip as an alias on the network interface

- On a remote workstation (or server), the pings respond, and ip1.20 is mapped with the MAC address of one of the 2 servers:

```
ping node1_ip_address; ping node2_ip_address; ping
virtip; arp –a
node1_ip_address       00-0c-29-0a-5c-fc
node2_ip_address       00-0c-29-26-44-93
virtip      00-0c-29-26-44-93
```

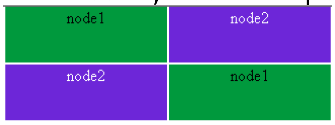
4.3.5 Test TCP load balancing on a virtual IP address

Farm module in the state ✓ `UP` (green) on the 2 servers node1, node2.

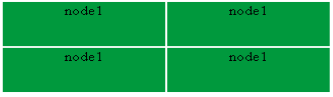
Same load balancing configuration in `userconfig.xml` as the previous test.

On a remote workstation:

1. Connect a browser to http://virtip:9010/safekit/mosaic.html. Click on the virtip link. node1, node2 respond



2. `safekit stop -m AM` on node2 (where AM is the module name). Reload the URL: node1 responds



Special command to check the load balancing bitmap for port 9010 on each node ✓ `UP` (green):

⇨ `safekit -r vip_if_ctrl -l`

An entry in the bitmap of 256 bits must be 1 on a single server.

Furthermore, the 256 bits are fairly distributed in the bitmaps of all servers ✓ `UP` (green) (if no definition of power inside `userconfig.xml`)

⇨ ✓ `UP` (green) on the 2 servers: load balancing of TCP sessions between node1, node2 when loading the URL

In the resources of the module, for node1 and node2: FarmProto 50%

Example of logs with node1 and node2:

In the logs of node1 and node2:

"farm membership: **node1 node2** (group FarmProto)"
"farm load: **128/256** (group FarmProto)"

128/256: 128 bits on 256 are managed by each server

`safekit -r vip_if_ctrl -l` on node1 and node2:

```
Bitmap 1:00000000:00000000:00000000:00000000:
ffffffff:ffffffff:ffffffff:ffffffff
Bitmap 2:ffffffff:ffffffff:ffffffff:ffffffff:
00000000:00000000:00000000:0000000
```

Bits are fairly distributed between both servers

⇨ ✗ `STOP` (red) on node2: TCP sessions served only by node1 when loading the URL

In the log of node1:

"farm membership: **node1** (group FarmProto)"
"farm load: **256/256** (group FarmProto)"

256/256: all the bits are managed by node1

`safekit -r vip_if_ctrl -l` on node1:

```
Bitmap 1:ffffffff:ffffffff:ffffffff:ffffffff:
ffffffff:ffffffff:ffffffff:ffffffff
```

All the bits are managed by node 1

4.3.6 Test split brain with a farm module

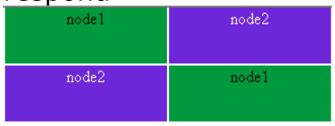
Split brain occurs in case of network isolation between SafeKit servers.

Farm module is ✓ `UP` (green) ✓ `UP` (green) on the servers node1 and node2.

Same configuration of load balancing in `userconfig.xml` as the previous test. To get the split brain, check in `userconfig.xml` that there are no checkers that can detect isolation: no `<interface check="on">` or `<ping>` checker

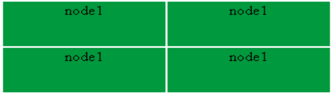
On the external workstation:

1. Connect a browser to http://virtip:9010/safekit/mosaic.html. Click on the virtip link. node1 and node2 respond



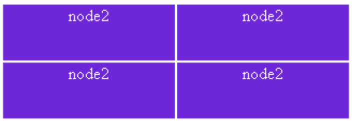
node1 node2
node2 node1

2. disconnect the network between node1 and node2. Depending on the location where the external console is, node 1 responds or node 2



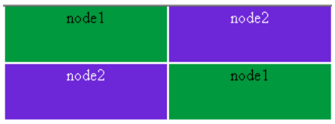
node1 node1
node1 node1

or



node2 node2
node2 node2

3. reconnect the network and connect to URL



node1 node2
node2 node1

Same special command as in the previous test to check the load balancing bitmap for port 9010 on each node ✓ `UP` (green)

➩ before split brain, state ✓ `UP` (green) ✓ `UP` (green):

In the resources of the module, for node1 and node2: FarmProto 50%.

In the logs of node1 and node2:

"farm membership: **node1 node2** (group FarmProto)"
"farm load: **128/256** (group FarmProto)"

128/256: 128 bits on 256 are managed by each server

`safekit -r vip_if_ctrl -l` on node1 and node2:

```
Bitmap 1:00000000:00000000:00000000:00000000:
ffffffff:ffffffff:ffffffff:ffffffff
Bitmap 2:ffffffff:ffffffff:ffffffff:ffffffff:
00000000:00000000:00000000:0000000
```

Bits are fairly distributed between both servers

➩ after isolation of servers, split brain:

In the resources of the module, for node1 and node2: FarmProto 100%.

In the log of node1:

"farm membership: **node1** (group FarmProto)"
"farm load: **256/256** (group FarmProto)"

256/256: all the bits are managed by node 1

`safekit -r vip_if_ctrl -l` on node1:

```
Bitmap 1:ffffffff:ffffffff:ffffffff:ffffffff:
ffffffff:ffffffff:ffffffff:ffffffff
```

in the log of node 2:

"farm membership: **node2** (group FarmProto)"
"farm load: **256/256** (group FarmProto)"

256/256: all the bits are managed by node 2

```
Bitmap 2:ffffffff:ffffffff:ffffffff:ffffffff:
         ffffffff:ffffffff:ffffffff:ffffffff
```

➩ after split brain when network is reconnected between ip1.1 and ip1.2, the same messages can be found in the log and the same bitmaps as those before split brain

Note: the default behavior of farm in situation of split brain is good. The recommendation is to put in `userconfig.xml` a monitoring network <lan></lan> where the virtual IP address is.

Note: In vmac_directed mode, the log messages and vip_if_ctrl output are different.

4.3.7 Test compatibility of the network with invisible MAC address (vmac_invisible)

4.3.7.1 Network prerequisite

A unicast MAC Ethernet address 5a-fe-xx-xx-xx-xx is associated with the virtual IP address of a farm module. It is never presented by SafeKit servers as source Ethernet address (invisible MAC). Switches cannot locate this address. When they follow a packet to the destination MAC address 5a-fe-xx-xx-xx-xx, they must broadcast the packet on all ports of the LAN or VLAN where the virtual IP address is (flooding). All servers in the farm therefore receive packets destined to the virtual MAC address 5a-fe-xx-xx-xx-xx.

Note that this prerequisite does not exist for a mirror module: see section 4.2.6 page 74

4.3.7.2 Server prerequisite

The packets are captured by Ethernet cards set in promiscuous mode by SafeKit. And the packets are filtered by the module kernel <vip> according the load balancing bitmap. To make a test, you need network monitor tool.

Network monitoring on Windows 2003 (CD2):

- install "Network Monitor Tools" in "Management and Monitoring Tools" (capture only packets in source or destination of the server)
- Start / Network Monitor then Capture Filter / Address Pairs / virtip then Capture / Start then "Stop and View" at the end of capture

Network monitoring on Linux:

- tcpdump host virtip: capture all network packets
- all servers are ✓ `UP` (green)
- the network monitoring is started on each server with a filter on virtip
- an external workstation sends a single ping to the virtual IP address with ping –n (or –c) 1 virtip
- result: 1 packet "ICMP: Echo: From ipconsole To virtip" sent and received by all servers
- result: there must be as many packets "ICMP: Echo Reply: To ipconsole From virtip" as there are servers ✓ `UP` (green)
- if it is not the case, check if options restrict the "port flooding" in switches and prevent the broadcast of "ICMP: Echo" to all servers
- be careful: the "port flooding" restriction in switches can occur after a certain number of flooding (time, number of KB flooded): the ping test must be repeated during several hours by creating flooding to the virtual IP address
- Note: to avoid network monitoring tools, an external Linux console can be used. The Linux ping prints duplicate packets coming from the 2 servers ✓ `UP` (green):

```
ping virtip
64 bytes from ip1.20 icmp_seq=1
64 bytes from ip1.20 icmp_seq=1 (DUP!)
64 bytes from ip1.20 icmp_seq=2
64 bytes from ip1.20 icmp_seq=2 (DUP!)...
```

This test may be carried out for several hours by storing the output of the ping in a file and then ensuring that there was (DUP!) all the time: `date > /tmp/ping.txt ; ping virtip >> /tmp/ping.txt`

4.3.8 Test farm module shutdown of a server ✓ UP (green)

- on Windows, check that the special procedure to stop modules at shutdown has been performed
- make a shutdown of a ✓ `UP` (green) server
- the other servers stay ✓ `UP` (green) and continue to run the application
- on timeout in the SafeKit console, the former server ✓ `UP` (green) becomes grey
- after reboot, check that shutdown of the OS has called a shutdown of the module

  "Action shutdown called by SYSTEM"
- Check that the `stop_both` script which stops the application has been executed with the message

  "Script `stop_both`"
- And check that the module has been completely stopped before stopping the server with the last message

  "End of stop"
- after reboot of the stopped server, if the module is started automatically at boot (`safekit boot status`), message in the log

  "Action start called at boot time"
- after start-up of the module on the stopped server, the module becomes ✓ `UP` (green) and it executes the `start_both` script which restarts the application on this server with the message in the log

  "Script `start_both`"

4.3.9 Test farm module power-off of a server ✓ UP (green)

- the other servers stay ✓ `UP` (green) and continue to run the application
- on timeout in the SafeKit console, the former server ✓ `UP` (green) becomes grey
- after reboot of the stopped server, if the module is started automatically at boot (`safekit boot status`), message in the log

  "Action start called at boot time"
- after reboot, message in the log

  "Previous halt unexpected"
- after start-up of the module on the stopped server, the module becomes ✓ `UP` (green) and it executes the `start_both` script which restarts the application on this server with the message in the log

  "Script `start_both`"

4.3.10 Continue your farm module tests with checkers

Go to section 4.4 page 86 for tests of checkers.

4.4 Tests of checkers common to mirror and farm

4.4.1 Test <errd>: checker of process with action restart or stopstart

In `userconfig.xml`:

```
<errd>
<proc name="appli.exe" atleast="1"

action="restart "

class="prim "/>
</errd>
```

- `name="appli.exe" atleast="1"`: at least one process "appli.exe" must run
- `class="prim"` (mirror module case): checker started on the server in state ✓ (green) (i.e. `PRIM` or `ALONE`), after `start_prim` script (stopped before `stop_prim`)
- `class="both"` (farm module case): checker started on all servers ✓ (green) `UP` after `start_both` script (stopped before `stop_both`)
- `action="restart"`: if `appli.exe` is not running, action `restart` which applies only scripts `stop_xx`; `start_xx`
- `action="stopstart"`: if `appli.exe` is not running, action `stopstart` which stops completely the module and then restarts it

Kill of process `appli.exe` on the server in ✓ (green) state. That is in states `PRIM` or `ALONE` for a mirror module; `UP` for a farm module:

- messages in the log:

  "event atleast on proc appli.exe"
  "Action restart|stopstart called by errd"

- the module becomes ✓ (magenta), respectively in state `PRIM`, `ALONE` or `UP`
- in the restart case, the module becomes ✓ (green), respectively in state `PRIM`, `ALONE` or `UP`
- in the stopstart case, the module becomes ✓ (green), respectively in state `SECOND`, `ALONE` or `UP`

  message in the log:

  "Action start called automatically"

  Note: a stopstart on ✓ (green) `PRIM` causes a failover

Repeat the test on the same server if it still runs the application (i.e. ✓ (green) in state `ALONE` or `UP`):

- with the default values of `maxloop="3"` and `loop_interval="24"` (`userconfig.xml` <service>)
- after 4 kills on the same server, the module becomes ✗ `STOP` (red)
- in the log, message before stopping:

  "Stopping loop"

4.4.2 Test <tcp> checker of the local application with action restart or stopstart

In userconfig.xml:

```
<tcp ident="id" when="prim ">
  <to addr="virtip" port="idport"
interval="10"

timeout="5" />
</tcp>
<failover>
<![CDATA[
tcpid_failure: if (tcp.id == down)
then stopstart();
]]>
</failover>
```

- the checker checks that the TCP application started on port idport responds to connection requests
- addr="virtip" port="idport" : TCP connections tested on IP address virtip and on TCP port idport
- interval="10" timeout="5" by default: test made every 10 seconds and with a timeout of 5 seconds
- when="prim" (mirror module case): checker is started on the server in state ✓ (green) (i.e. PRIM or ALONE), after the start_prim script (stopped before stop_prim)
- when="both" (farm module case): checker is started on all servers in state ✓ (green) UP, after the start_both script (stopped before stop_both)
- action restart(): Default failover rule; if the local TCP connection fails, action restart which runs only scripts stop_xx ; start_xx
- action stopstart(): if the local TCP connection fails, action stopstart which stops completely the module and then restarts it

Stop the application listening on port idport on the server in state ✓ (green). That is in states PRIM or ALONE for a mirror module, UP for a farm module:

- messages in the log:

  "Resource tcp.id set to down by tcpcheck"
  "Action restart|stopstart from failover rule tcpid_failure "
- the module becomes ✓ (magenta), respectively in state PRIM, ALONE or UP
- in the restart case, the module becomes ✓ (green), respectively in state PRIM, ALONE or UP
- in the stopstart case, the module becomes ✓ (green), respectively in state SECOND, ALONE or UP.

  message in the log:

  "Action start called automatically"

  Note: a stopstart on ✓ (green) PRIM causes a failover.

Repeat the test on the same server if it still runs the application (i.e. ✓ (green) in state ALONE or UP):

- with the default values of maxloop="3" loop_interval="24" (userconfig.xml <service>)
- after 4 stops of the application on the same server, the module becomes ✗ STOP (red)
- in the log, message before stopping:

  "Stopping loop"

4.4.3 Test <tcp> checker of an external service with action wait

In userconfig.xml:

```
<tcp ident="id" when="pre">
  <to addr="ip.external" port="idport"
interval="10"

timeout="5" />
</tcp>
<failover>
<![CDATA[
tcpid_failure: if (tcp.id== down) then
wait();
]]>
</failover>
```

- the checker checks that the external TCP service (ip.external, idport) responds to connection requests
- interval="10" timeout="5" by default: test made every 10 seconds and with a timeout of 5 seconds
- when="pre": started at the beginning of module start-up after prestart script (and stopped before poststop)
- if the TCP connection fails, the checker sets the resource tcp.id to down. The failover rule on the TCP checker runs the stopwait action which stops the application and puts the module in the state WAIT, waiting for tcp.id reset to up by the checker

Stop the external TCP service (ip.external, idport), on the server in ✓ (green) state. That is in state PRIM, ALONE or SECOND for a mirror module, UP for a farm module:

- messages in the log:

  "Resource tcp.id set to down by tcpcheck"
  "Action wait from failover rule tcpid_failure"

  Note: a wait on ✓ (green) PRIM causes a failover
- in all cases, the server becomes ✗ WAIT (red) on the server

Restart the external TCP process and services:

- messages in the log

  "Resource tcp.id set to up by tcpcheck"
  "Transition WAKEUP from failover rule Implicit_WAKEUP"
- the module restarts on the server and becomes ✓ (green), respectively in state SECOND, ALONE, SECOND or UP

Repeat the test on the same server:

- with the default values of maxloop="3" loop_interval="24" (userconfig.xml <service>)
- after 4 restarts of on the same server, the module becomes ✗ STOP (red)
- in the log, message before stopping:

  "Stopping loop"

Note: This test allows testing of connectivity to an external service. But if the external service is down or is unreachable on all servers, all servers are in state ✗ WAIT (red) and the application is unavailable

4.4.4 Test <interface check="on"> on a local network interface and with action wait

In `userconfig.xml`:

```
<vip>
 <interface_list>
  <interface check="on">
      <!--
         definition of a virtual IP
address
         on the network default
       -->
  </interface>
 </interface_list>
</vip>
```

Default failover rule = wait

- A checker checks that the Ethernet cable is connected in the interface of the `ip.0` network where the virtual IP address is set
- If the cable is disconnected, the checker updates the resource `intf.ip.0` to down. The failover rule on interface checkers runs the stopwait action which stops the application and puts the module in the `WAIT` state waiting for intf.ip.0 reset to up by the checker.

Note: do not use `check="on"` on bonding or teaming interface because these interfaces bring their own failover mechanisms from interface to interface

Unplug the Ethernet cable from ip.0 network on the server in ✓ (green) state. That is in state `PRIM`, `ALONE` or `SECOND` for a mirror module, `UP` for a farm module:

- messages in the log:

  "Resource intf.ip.default set to down by intfcheck"

  "Action wait from failover rule interface_failure"

  "Transition `WAIT`_TR from failover rule interface_failure"

  Note: a wait on ✓ (green) `PRIM` causes a failover
- in all cases, the module becomes ✗ `WAIT` (red) on the server

Plug the cable again:

- messages in the log

  "Resource intf.ip.0 set to up by intfcheck"
  "Transition WAKEUP from failover rule Implicit_WAKEUP"
- the module restarts on the server and becomes ✓ (green), respectively in state `SECOND`, `ALONE`, `SECOND` or `UP`

Repeat the test on the same server:

- with the default values of `maxloop="3" loop_interval="24"` (`userconfig.xml` <service>)
- after 4 restarts on the same server, the module becomes ✗ `STOP` (red)
- in the log, message before stopping:

  "Stopping loop"

  Note: disabling the interface (instead of unplugging the ethernet cable) leads to ✗ `STOP` (red). The reason is that the module cannot start (or restart) without local IP address.

4.4.5 Test <ping> checker with action wait

In userconfig.xml:

```
<ping ident="id" when="pre">
  <to addr="ip.device" interval="10"

timeout="5"/>
</ping>
Default failover rule = wait
```

- the checker checks that the external device (ex.: a router) with address ip.device responds to ping
- interval="10" timeout="5" by default: test made every 10 seconds and with a timeout of 5 seconds
- when="pre": started at the beginning of module start-up after prestart script (and stopped before poststop)
- if the ping does not respond, the checker sets the resource ping.id to down. The failover rule on ping checker runs the stopwait action which stops the application and puts the module in the WAIT state, waiting for ping.id reset to up by the checker.

Break the link between the pinged external device and the server the server in ✓(green) state. That is in state PRIM, ALONE or SECOND for a mirror module, UP for a farm module:

- messages in the log:

  "Resource ping.id set to down by pingcheck"
  "Action wait from failover rule ping_failure"

  Note: a wait on ✓ (green) PRIM causes a failover
- in all cases, the module becomes ✗ WAIT (red) on the server

Restore the network connection:

- messages in the log

  "Resource ping.id set to up by pingcheck"
  "Transition WAKEUP from failover rule Implicit_WAKEUP"
- the module restarts on the server and becomes ✓ (green), respectively in state SECOND, ALONE, SECOND or UP

Repeat the test on the same server:

- with the default values of maxloop="3" loop_interval="24" (userconfig.xml <service>)
- after 4 restarts on the same server, the module becomes ✗ STOP (red)
- in the log, message before stopping:

  "Stopping loop"

Note: this test allows testing of connectivity from the server to the network. But if the external device is down and if the ping fails on all servers, all servers are in ✗ WAIT (red) and the application is unavailable.

4.4.6 Test <module> checker with action wait

In `userconfig.xml` of module X, test of another module othermodule:

```
userconfig.xml of module X:

<module name="othermodule">
  <to addr="ip" interval="10"
timeout="5"/>
</module>
```

- the checker checks the module othermodule on its virtual IP address ip
- `interval="10" timeout="5"` by default: test made every 10 seconds and with a timeout of 5 seconds

If the module othermodule is not started, the module X stay in the `WAIT` state waiting for its restart

The module X makes a stopstart when the module othermodule is restarted

Note: if the module X is a mirror module using file replication and because of rule `notuptodate_server`, you may experience a wrong behavior with module X blocked in a `WAIT` state, if the stopstart action happens when X in the transition `SECOND` to `ALONE`

Stop the module othermodule. And start the module X on all servers:

- messages in the log of module X

  "Resource module.othermodule_ip set to down by modulecheck
  "Action wait from failover rule module_failure"
- the module X becomes ✗ `WAIT` (red) on all servers

Start the module othermodule:

- messages in the log of module X

  "Resource module.othermodule_ip set to up by modulecheck"
  "Transition WAKEUP from failover rule Implicit_WAKEUP"
- the module X starts on all servers in ✓ (green)

Make `safekit restart -m othermodule`

- messages in the log of module X:

  "Action stopstart called by modulecheck"
- the module X stops and then restarts

Repeat the test on the same server:

- with the default values of `maxloop="3" loop_interval="24"` (`userconfig.xml` <service>)
- after 4 restarts on the same server, the module becomes ✗ `STOP` (red)
- in the log, message before stopping:

  "Stopping loop"

4.4.7 Test <custom> checker with action wait

In `userconfig.xml`:

```
<custom ident="id" when="pre"

exec="customscript" >
</custom>
```

- script `SAFE/module/<name>/bin/customscript` is a custom checker: a loop with a test on a resource
- `when="pre"`: custom checker started on all servers ✓ `PRIM`, `ALONE`, `SECOND` or `UP` (green) after prestart script (stopped before poststop)

Manage the resource custom.id to perform the action:

- in the script customscript:
  on error: `SAFE/`safekit set -r custom.id–v down –i customscript

  on success: `SAFE/`safekit set -r custom.id–v up –i customscript
- in `userconfig.xml`:
  <failover>
  <![CDATA[
  customid_failure: if (custom.id == down) then wait();
  ]]>
  </failover>

- if the custom checker sets the resource to down, action wait which stops completely the module and restarts it in the state ✗ `WAIT` (red), waiting for the resource reset to up by the custom checker

Cause the error tested by custom checker on the server in state ✓ (green). That is in state `PRIM`, `ALONE` or `SECOND` for a mirror module; `UP` for a farm module:

- messages in the log:

  "Resource custom.id set to down by customscript"
  "Action wait from failover rule customid_failure"
  "Transition `WAIT`_TR from failover rule customid_failure"

  Note: a wait on ✓ (green) `PRIM` causes a failover
- in all cases, the module becomes ✗ `WAIT` (red) on the server

Fix error tested by custom checker:

- messages in the log

  "Resource custom.id set to up by customscript"
  "Transition WAKEUP from failover rule Implicit_WAKEUP"
- the module restarts on the server and becomes ✓ (green), respectively in state `SECOND`, `ALONE`, `SECOND` or `UP`

Repeat the test on the same server:

- with the default values of `maxloop="3" loop_interval="24"` (`userconfig.xml` <service>)
- after 4 restarts on the same server, the module becomes ✗ `STOP` (red)
- in the log, message before stopping:

  "Stopping loop"

4.4.8 Test <custom> checker with action restart or stopstart

4.4.8.1 Action through a failover rule

In `userconfig.xml`:

```
<custom ident="id" when="prim "
exec="customscript" >
</custom>
```

- script customscript `SAFE/module/<name>/bin/customscript` is a custom checker: loop with a test on the application integrated in the scripts

- `when="prim"` (mirror module case): custom checker started on the server ✓ `PRIM` or `ALONE` (green) after `start_prim` script (stopped before stop_prim)

- `when="both"` (farm module case): custom checker started on all servers ✓ `UP` (green) after `start_both` script (stopped before `stop_both`)

Manage the resource custom.id to perform the action:

- in the script customscript:

  on error: safekit set -r custom.id–v down –i customscript

  on success: safekit set -r custom.id–v up –i customscript

```
in userconfig.xml:

<failover>
<![CDATA[
customid_failure: if (custom.id ==
down) then restart ();
]]>
</failover>
or
<failover>
<![CDATA[
customid_failure: if (custom.id ==
down) then stopstart ();
]]>
</failover>
```

Cause the error tested by custom checker on the server in state ✓ (green). That is in state `PRIM`, `ALONE` or `SECOND` for a mirror module; `UP` for a farm module:

- messages in the log

  "Resource custom.id set to down by customscript"

  "Action restart from failover rule customid_failure"

  "Transition RESTART from failover rule customid_failure"

- the module becomes ✓ (magenta), respectively in state `PRIM`, `ALONE` or `UP`

- in the restart case, the module becomes ✓ (green), respectively state `PRIM`, `ALONE` or `UP`

- in the stopstart case, the module becomes ✓ (green), respectively in state `SECOND`, `ALONE` or `UP`

  message in the log

  "Action start called automatically"

  Note: a stopstart on ✓ (green) `PRIM` causes a failover

Repeat the test on the same server if it still runs the application (i.e. ✓ (green) in state `ALONE` or `UP`):

- with the default values of `maxloop="3"` `loop_interval="24"` (`userconfig.xml` <service>)

- after 4 restarts on the same server, the module becomes ✗ `STOP` (red)

- in the log, message before stopping: "Stopping loop"

4.4.8.2 Action through a command in the custom checker

In `userconfig.xml`:

```
<custom ident="id" when="prim "
            exec="customscript" >
</custom>
```

- script `SAFE/module/<name>/bin/customscript` is a custom checker: loop with a test on the application integrated in the scripts
- `when="prim"` (mirror module case): custom checker started on the server ✓ `PRIM` or `ALONE` (green) after `start_prim` script (stopped before stop_prim)
- `when="both"` (farm module case): custom checker started on all servers ✓ `UP` (green) after `start_both` script (stopped before `stop_both`)

On error, run command `restart|stopstart`:

- in the script customscript:
  on error: safekit restart -i customscript

  or

  safekit stopstart -i customscript
- action restart: run only scripts `stop_xx` ; `start_xx`
- action stopstart: stop completely the module and then restart it

Cause the error tested by custom checker on the server in state ✓ (green). That is in state `PRIM`, `ALONE` or `SECOND` for a mirror module, `UP` for a farm module:

- messages in the log

  "Action restart called by customscript"

  Or

  "Action stoptart called by customscript"
- the module becomes ✓ (magenta), respectively in state `PRIM`, `ALONE` or `UP`
- in the restart case, the module becomes ✓ (green), respectively in state `PRIM`, `ALONE` or `UP`
- in the stopstart case, the module becomes ✓ (green), respectively in state `SECOND`, `ALONE` or `UP`

  message in the log

  "Action start called automatically"

  Note: a stopstart on ✓ (green) `PRIM` causes a failover

Repeat the test on the same server if it still runs the application (i.e. ✓ (green) in state `ALONE` or `UP`):

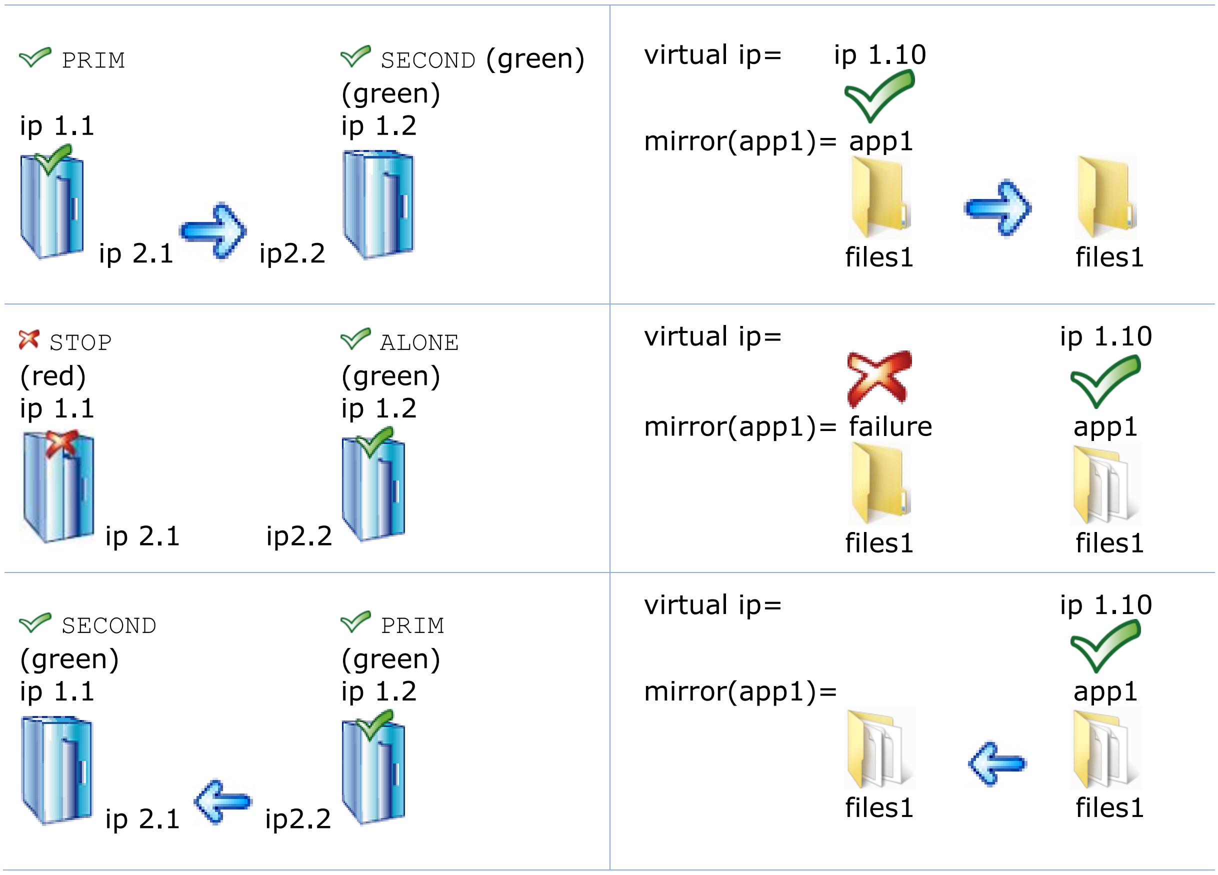
- with the default values of `maxloop="3"` `loop_interval="24"` (`userconfig.xml` <service>)
- after 4 restarts on the same server, the module becomes ✗ `STOP` (red)
- in the log, message before stopping: "Stopping loop"

Note: on a direct action in the custom checker, the loop counter is incremented if `-i identity` is passed to the command restart or stopstart. Without identity, SafeKit considers the command is as an administrative operation. The counter is reset and there is no stop after 4 restarts.

5.Mirror module administration

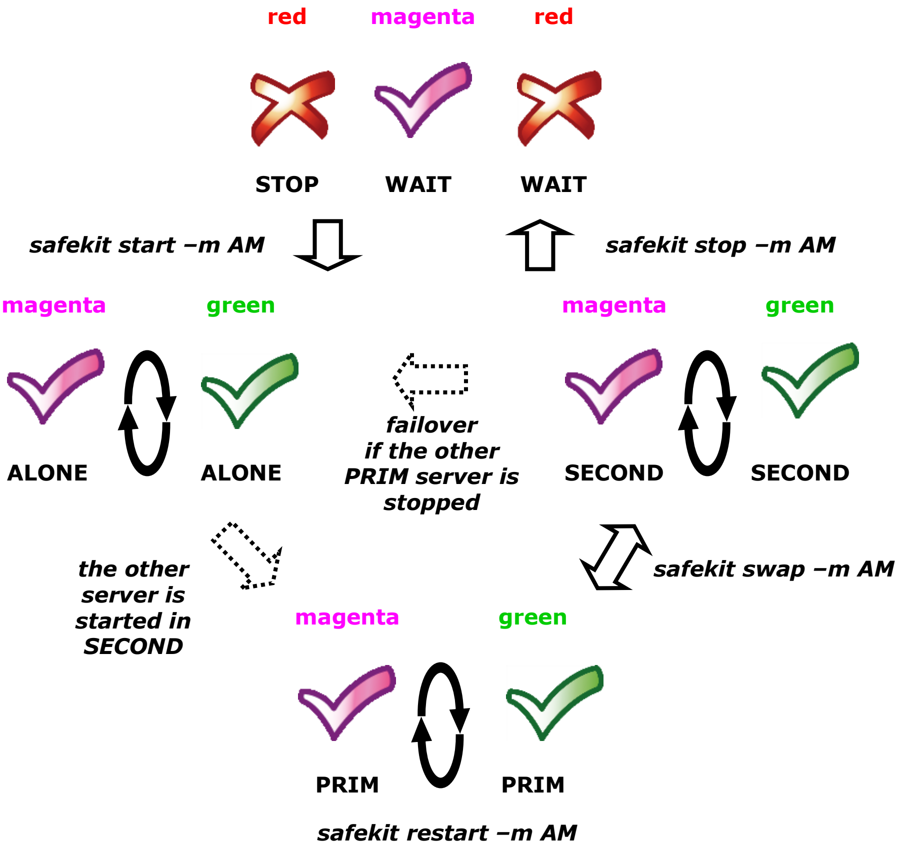
5.1 Operating mode of a mirror module

To test a mirror module, see section 4.2 page 72



5.2 State automaton of a mirror module (STOP, WAIT, ALONE, PRIM, SECOND - red, magenta, green)

To analyze a problem, see section 7 page 109



✗ STOP (red):	module stopped
✓ WAIT (magenta):	in start-up phase
✗ WAIT (red):	blocked because of a resource="down"
✓ ALONE (magenta):	primary without secondary; executing application scripts (start_prim or stop_prim)
✓ ALONE (green):	primary without secondary; stable; application is started
✓ SECOND (magenta):	in the process of reintegrating data from the primary before becoming secondary
✓ SECOND (green):	secondary with primary; stable; ready to resume the application and to become primary
✓ PRIM (magenta):	primary with secondary; executing application scripts (start_prim or stop_prim)
✓ PRIM (green):	primary with secondary; stable; application is started

5.3 First start-up of a mirror module (prim command)

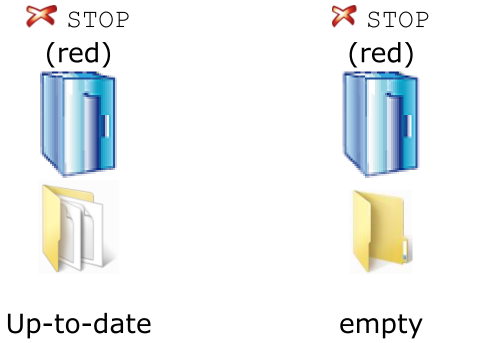
At first start-up of a mirror module, if both servers are started with the start command, both go into ✘ `WAIT` (red) state with the message "Data may be not uptodate for replicated directories (wait for the start of the remote server)" in the log.

At first start-up of a mirror module, use the special prim command on the server with the up-to-date directory, and the second command on the other one. Data is synchronized from the primary server to the secondary one.

For next start-up, use the start command on both servers.

1. initial state

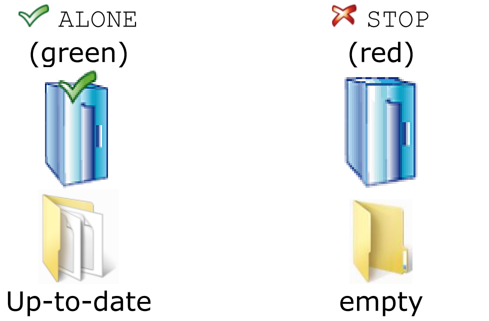
- the mirror module has just been configured with a new directory to replicate between server 1 and server 2
- server 1 has the up-to-date directory
- server 2 has an empty directory



2. command prim on server 1

- use the special prim command to force server 1 to become primary
- for following start-ups, always prefer start: see section 5.5 page 99
- message in the log:

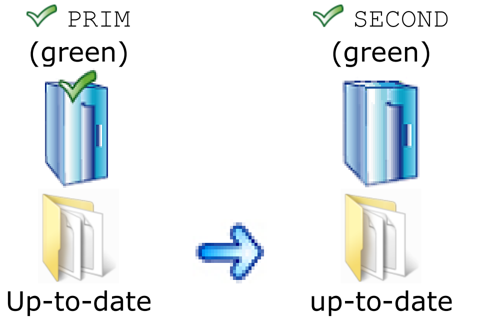
"Action prim called by web@<IP>/SYSTEM/root"



3. command second on server 2

- start the other server as secondary
- the secondary reintegrates replicated directory from primary
- message in the log:

"Action second called by web@<IP>/SYSTEM/root"



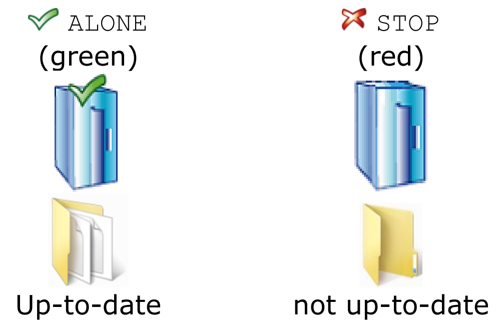
5.4 Different reintegration cases (use of bitmaps)

To optimize file reintegration, different cases are considered:

1. The module must have completed the reintegration (on the first start of the module, it runs a full reintegration) before enabling the tracking of modification into bitmaps
2. If the module was cleanly stopped on the server, then at restart of the secondary, only the modified zones of modified files are reintegrated, according to a set of modification tracking bitmaps.
3. If the server crashed (power off) or was incorrectly stopped (exception in nfsbox replication process), or if files have been modified while SafeKit was stopped, the modification bitmaps are not reliable, and are therefore discarded. All the files bearing a modification timestamp more recent than the last known synchronization point minus a grace delay (typically one hour) are reintegrated.
4. A call to the special `second fullsync` command triggers a full reintegration of all replicated directories on the secondary when it is restarted.

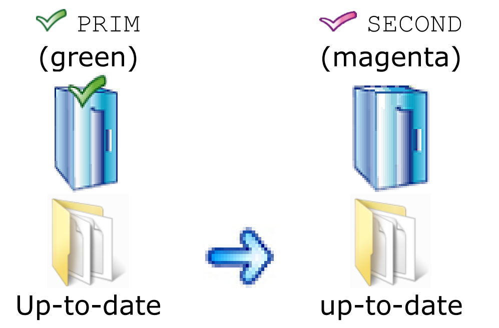
1. secondary server2 has been stopped

➩ data is desynchronized



2. start command on server 2

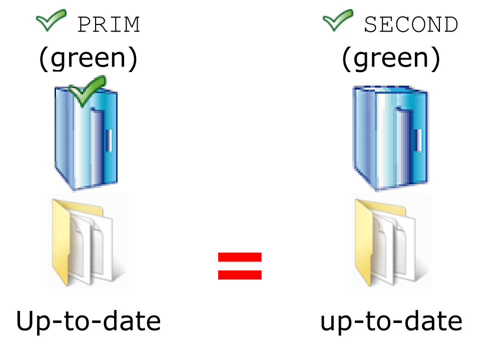
➩ data is reintegrated with bitmap optimization (see above)



3. end of reintegration

➩ data is the same on both servers

➩ only modifications inside files are replicated with a real-time synchronous replication



The replication system also keeps track of the last date on which data was synchronized on each node. This synchronization date, named `synctimestamp`, is assigned at the end of the reintegration and changes in the ✓ `PRIM` (green) and ✓ `SECOND` (green) states. When the module is stopped on the secondary node and then restarted, the `synctimestamp` is one of the reintegration criteria: all files modified around this date are potentially out of date on the secondary and must be reintegrated. Since SafeKit 7.4.0.50, the synchronization date is also used to implement an additional security. When the difference between the synchronization date stored on the primary and on the secondary is greater than 90 seconds, the replicated data is considered unsynchronized in its entirety. The reintegration is interrupted with the following message in the module log:

```
| 2021-08-06 08:40:20.909224 | reintegre | E | Automatic synchronization
cannot be applied due to an abnormal delta between the dates of the last
synchronization
```

If the administrator considers that the server is valid, he can force the start in secondary with full synchronization of the data, by executing the command: `safekit second fullsync -m AM`.

5.5 Start-up of a mirror module with the up-to-date data (✗ STOP (red) - ✗ WAIT (red))

SafeKit determines which server must start as primary or not. SafeKit retains the information on the server with the up-to-date replicated directories. To take advantage of this feature, use the command start and NOT the command prim

1. initial state ⇨ server1 is primary `ALONE` ⇨ directories are up-to-date on this server ⇨ the module is stopped on server 2 ⇨ server 2 has desynchronized replicated directories	✓ `ALONE` (green) Up-to-date	✗ `STOP` (red) not up-to-date
2. command stop on server 1 ⇨ stop of the server with the up-to-date directories	✗ `STOP` (red) Up-to-date	✗ `STOP` (red) not up-to-date

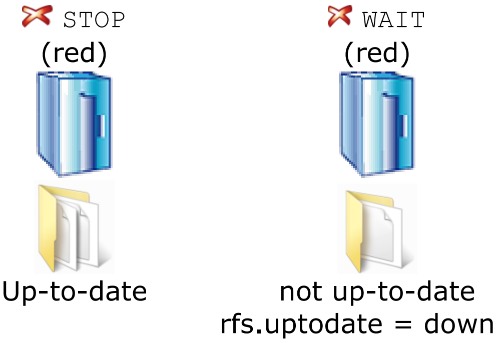
3. command start on server 2

➩ the module is put in the `WAIT` state waiting for the start of the other server and within its log of messages:

"Data may be not uptodate for replicated directories (wait for the start of the remote server)"
"Action wait from failover rule notuptodate_server"
"If you are sure that this server has valid data, run safekit prim to force start as primary"

➩ in this case, you must start server1 to resynchronize data of server2

➩ if you really want to sacrifice the up-to-date data and start server 2 as primary with the data not up-to-date: issue a stop command then a prim command on server 2



See also 5.9 "Prim command fails: why? (command primforce)" page 104

5.6 Degraded replication mode (✓ ALONE (green) degraded)

If the replication process nfsbox fails on the primary server (for instance because of an unrecoverable replication problem), the application is not swapped on the secondary server

The primary server goes to the `ALONE` state in a degraded replication mode.

Degraded is displayed in the web console/✓ Control/ under the `ALONE` server. A "Resource rfs.degraded set to up by nfsadmin" message is emitted in the log.

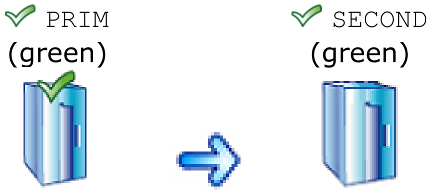
`safekit state -v -m AM` returns resource rfs.degraded up (replace AM by the module name)

The primary server continues in `ALONE` state with a nfsbox process which does not replicate anymore.

You must stop and start the `ALONE` server to come back to a `PRIM` – `SECOND` state with replication

1. initial state

➩ the mirror is in a stable state server 1 ✓ `PRIM` (green) – server 2 ✓ `SECOND` (green)



2. failure of replication process nfsbox on server 1

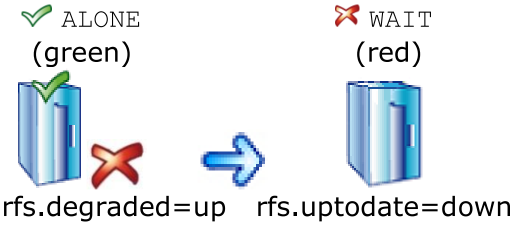
- server 1 becomes ✓ `ALONE` (green) degraded with the message in its log

  "Resource rfs.degraded set to up by nfsadmin".

  `safekit state -v AM` returns resource rfs.degraded=up (where AM is the module name)
- server 1 `ALONE` continues to execute the application without replication
- server 2 is in ✗ `WAIT` (red) waiting for the replication process with the message in its log

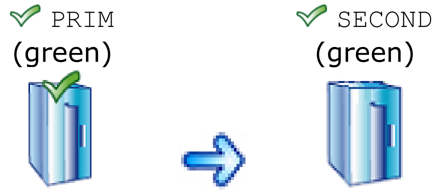
  "Action wait from failover rule degraded_server"

  and with rfs.uptodate=down



3. come back to replication

- administrator makes stop command and start command on server 1 `ALONE`
- the nfsbox replication process is restarted on server 1
- server 2 reintegrates replicated directories before becoming ✓ `SECOND` (green)
- server 1 becomes ✓ `PRIM` (green)



5.7 Automatic or manual failover (failover="off" - ✗ STOP (red) - ✗WAIT (red))

Automatic or manual failover on the secondary server is defined in `userconfig.xml` by `<service mode="mirror" failover="on"|"off">`. By default, if the parameter is not defined, `failover="on"`

The `failover="off"` mode is useful when the failover must be controlled by an administrator. This mode ensures that an application runs always on the same primary server whatever operations are made on the server (reboot, temporary stop of the module for maintenance...). Only an explicit administrative action (`prim` command) may promote the other server as primary.

Note: Failover mode could be set dynamically on a running cluster with the `safekit failover on|off -v AM` line command (replace AM by the module name) or from the web console in action menu.

1. initial state ➾ the mirror is in a stable state server 1 ✓ `PRIM` (green) – server 2 ✓ `SECOND` (green)	✓ `PRIM` (green)	✓ `SECOND` (green)
2. restart with failover="on" ➾ if server 1 former `PRIM` fails and stops, server 2 becomes automatically `ALONE` (default mode)	✗ `STOP` (red)	✓ `ALONE` (green)
3. behavior with failover="off" ➾ if server 1 former `PRIM` fails and stops, server 2 goes to ✗ `WAIT` (red) state with message in its log "Failover-off configured" "Action stopstart called by failover-off" "Transition STOPSTART from failover-off" "Local state `WAIT` red " ➾ the administrator in this situation can restart server 1: the mirror restarts in its former stable state server 1 ✓ `PRIM` (green) – server 2 ✓ `SECOND` (green) ➾ the administrator can decide to force server 2 to become primary with the command: `stop` then `prim` on server 2	✗ `STOP` (red)	✗ `WAIT` (red)

See also section 5.9 page 101

5.8 Default primary server (automatic swap after reintegration)

After reintegration at failback, a server becomes by default secondary. The administrator may choose to swap the application back to the reintegrated server at an appropriate time with the `swap` command. This is the default behavior when `userconfig.xml` <service> is defined without the defaultprim variable

If the application must automatically swap back to a preferred server after reintegration, specify a defaultprim server in `userconfig.xml`: <service mode="mirror" defaultprim="hostname server 1">

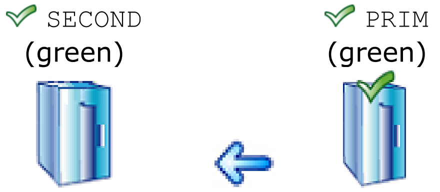
1. **initial state**

- server 1 (former `PRIM`) fails and stops
- server 2 secondary becomes automatically `ALONE`

✗ STOP
(red)
✓ ALONE
(green)

2. **reintegration without defaultprim**

- server 1 is restarted with command start
- it reintegrates replicated directories and then becomes secondary
- an administrator can swap the primary to server 1 with the command `swap` in a timely manner
- swap stops the application on server 2 and restarts it on server 1

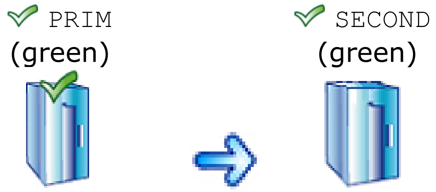


3. **reintegration with defaultprim="hostname server 1"**

- server 1 in ✗ `STOP` (red) at step 1 (initial state) is restarted by command start
- it reintegrates replicated directories
- just after reintegration, an automatic swap is made on server 1 with the message in its log:

  "Transition SWAP from defaultprim"
  "Begin of Swap"

- the application is then automatically stopped on server 2 and restarted on server 1
- at the end, server 1 is `PRIM`

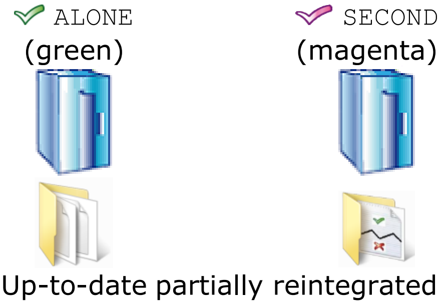


5.9 Prim command fails: why? (command primforce)

A prim command may fail to start a server as primary: after trying a start-up, the server goes back to `STOP` (red).

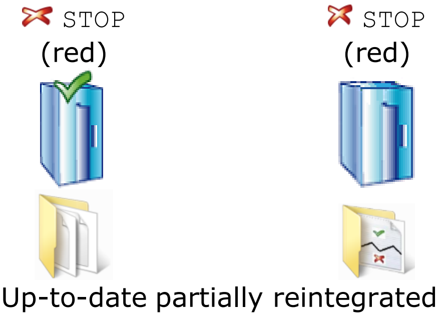
1. initial state

- server 1 `ALONE` has the up-to-date directory
- server 2 is in the process of reintegrating files from server 1



2. command stop on server 2 then on server 1

- stop of server 2 during its reintegration: stop of server 2 can be made while a file that is half copied (corrupted file)
- server 1 is also stopped



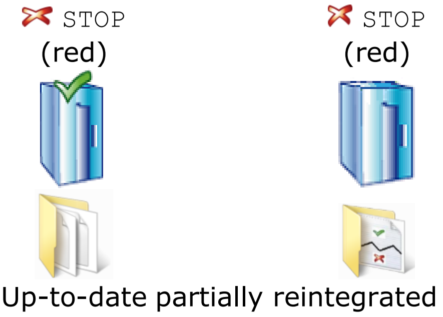
3. command prim on server 2

- fails with messages in the log described above

"Data may be inconsistent for replicated directories (stopped during reintegration)"
"If you are sure that this server has valid data, run safekit primforce to force start as primary"

- in this case, you must start server 1 with start command or prim command. And to restart server 2 with start command to finish reintegration of files. While server 2 is not in the state ✓ `SECOND` green, its data may be corrupted
- if you absolutely want to start as primary on server 2 partially reintegrated and with data potentially corrupted, use the command `safekit primforce -m AM` on server 2 (command line only, where AM is the module name). Message in the log:

  "Action primforce called by SYSTEM/root"



command prim fails
because data
may be corrupted

Note: The `safekit primforce -m AM` command forces a full reintegration of replicated directories on the secondary when it is restarted.

6. Farm module administration

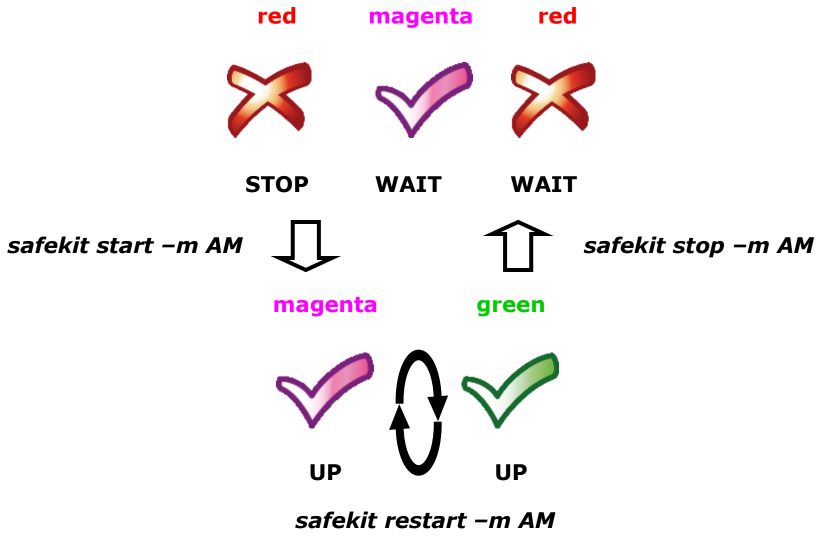
6.1 Operating mode of a farm module

To test a farm module, see section 4.3 page 79.

✓ UP (green) ip 1.1 ✓	✓ UP (green) ip 1.2 ✓	✓ UP (green) ip 1.3 ✓	virtual ip= farm(app2)=	ip 1.20 ✓ app2	ip 1.20 ✓ app2	ip 1.20 ✓ app2
✓ UP (green) ip 1.1 ✓	✗ STOP (red) ip 1.2 ✗	✓ UP (green) ip1.3 ✓	virtual ip= farm(app2)=	ip 1.20 ✓ app2	✗	ip 1.20 ✓ app2
✓ UP (green) ip 1.1 ✓	✓ UP (green) ip 1.2 ✓	✓ UP (green) ip1.3 ✓	virtual ip= farm(app2)=	ip 1.20 ✓ app2	ip 1.20 ✓ app2	ip 1.20 ✓ app2

6.2 State automaton of a farm module (STOP, WAIT, UP - red, magenta, green)

To analyze a problem, see section 7 page 109



✗ STOP (red):	module stopped
✓ WAIT (magenta):	in the start-up phase
✗ WAIT (red):	blocked because of a resource="down"
✓ UP (magenta):	executing application scripts (start_both or stop_both)
✓ UP (green):	stable with application started

Note: This is also the state automation of a light module. A light module is identified by <service mode="light"> in userconfig.xml file under SAFE/modules/AM/conf (where AM is the module name). The light type corresponds to a module that runs on a server without synchronizing with other servers (as can-do mirror or farm modules). A light module includes the start and stop of an application as well as the SafeKit checkers that can detect errors.

6.3 Start-up of a farm module

Use the start command on each server running the module. An example with a farm of 2 servers is presented below.

1. initial state ⇨ the farm module has just been configured on server 1 and server 2	✗ STOP (red)	✗ STOP (red)
2. command start on server 1 and server 2 ⇨ message in the log of both servers: "farm membership: **node1 node2** (group FarmProto)" "farm load: **128/256** (group FarmProto)" "Local state UP green" ⇨ resource of the module instance on both servers: FarmProto 50%	✓ UP (green) ✓	✓ UP (red)

7. Troubleshooting

7.1 Connection issues with the web console

If you encounter problems for connecting to the SafeKit web console to SafeKit node, such as no reply or connection error, run the following checks and procedures:

Then, it may be necessary to reload the console into the browser.

7.1.1 Browser check

For the web browser, check:

- ✓ that it is a supported browser and its level (Chrome works better than Internet Explorer in many environments)
- ✓ change the proxy settings for direct or indirect connection to the server

- ✓ with Internet Explorer, change the security settings (add the URL into the trusted zones)
- ✓ clear the browser's state on upgrade as described below
- ✓ that the web console and the server are at the same level (backward compatibility may not be fully preserved)

7.1.2 Browser state clear

1. Clear the browser cache

   A quick way to do this is a keyboard shortcut that works on IE, Firefox, and Chrome. Open the browser to any web page and hold CTRL and SHIFT while tapping the DELETE key. (This is NOT CTRL, ALT, DEL). The dialog box will open to clear the browser. Set it to clear everything and click Clear Now or Delete at the bottom

2. Clear the browser SSL cache if HTTPS is used

   Look at advanced settings for the browser and search for SSL cache.

Finally close all windows for the browser, stop the browser process still running in the background if necessary, and re-open it fresh to test what wasn't working for you previously.

7.1.3 Server check

On each SafeKit cluster node check:

- ✓ the firewall

  If this has not yet been done, run the `SAFE/bin/firewallcfg add` command which configures the operating system firewall. For other firewalls, add an exception to allow connections between the web browser and the server. For details, see section 10.3 page 156.

- ✓ the web server configuration

  Since SafeKit 7.5, HTTP access to the web console requires authentication. If it has not yet been done, run the `SAFE/bin/webservercfg -passwd pwd` to initialize (or reinitialize) this configuration with the password of the user `admin`. For details, see 11.2.1 page 179.

- ✓ the network and the server availability
- ✓ the `safeadmin` and `safewebserver` services

  They must be started.

- ✓ the SafeKit cluster configuration

  Run the command `safekit cluster confinfo` (see section 9.3 page 144). This command must return on all nodes, the same list of nodes and the same value for the configuration signature. If not, reapply the cluster configuration on all nodes (see section 12.2 page 231).

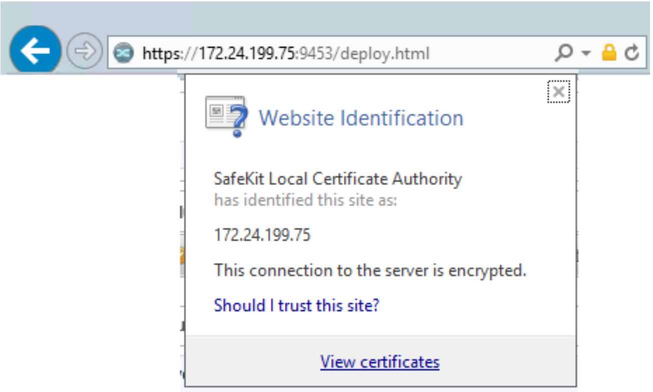
7.2 Connection issues with the HTTPS web console

If you encounter problems for connecting the secure SafeKit web console to SafeKit nodes, you can run the following checks and procedures:

7.2.1 Check server certificates

The SafeKit web console connects to a SafeKit node that is identified by a certificate. To get the SafeKit node certificate content with Internet Explorer or Chrome, run the following:

1. Click on the lock next to the URL to open the security report
2. Click on the View certificates link. It opens a window that displays the certificate content



https://172.24.199.75:9453/deploy.html

Website Identification

SafeKit Local Certificate Authority
has identified this site as:

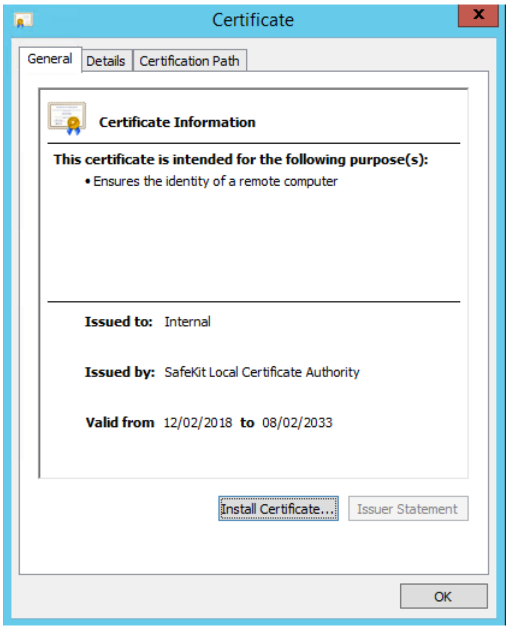
172.24.199.75

This connection to the server is encrypted.

Should I trust this site?

View certificates

3. Check the issuer that must be the appropriate certification authority
4. Check the validity date and the workstation date. If necessary, change the workstation date
5. Check the validity date. If the certificate is expired, you must renew. For certificate generated with the SafeKit PKI, see section 11.6.2 page 221



Certificate

Certificate Information

This certificate is intended for the following purpose(s):

- Ensures the identity of a remote computer

Issued to: Internal

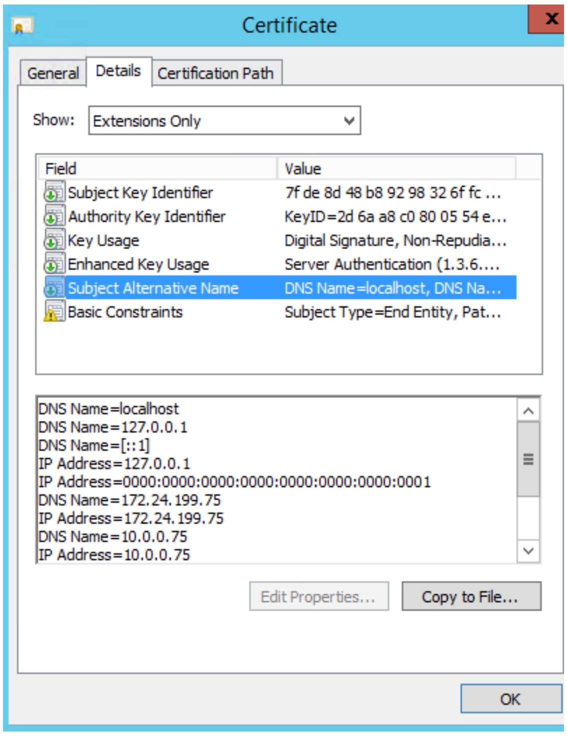
Issued by: SafeKit Local Certificate Authority

Valid from 12/02/2018 to 08/02/2033

6. Click on Details tab
7. Select Subject Alternate Name field. Its content is displayed into the bottom panel. The location set into the URL for connecting the SafeKit web console must be included into this list. Change the URL if necessary
8. localhost and 127.0.0.1 must be present
9. The address value for the node, set into the SafeKit cluster configuration, must be one of the values listed. If it is not, change the cluster configuration as described in 12.2 page 231.

   When using DNS name, you must use lower case.

   ⚠ With SafeKit <= 7.5.2.9, the server’s name must be included.



Certificate

Show: Extensions Only

Field	Value
Subject Key Identifier	7f de 8d 48 b8 92 98 32 6f fc ...
Authority Key Identifier	KeyID=2d 6a a8 c0 80 05 54 e...
Key Usage	Digital Signature, Non-Repudia...
Enhanced Key Usage	Server Authentication (1.3.6....
Subject Alternative Name	DNS Name=localhost, DNS Na...
Basic Constraints	Subject Type=End Entity, Pat...

```
DNS Name=localhost
DNS Name=127.0.0.1
DNS Name=[::1]
IP Address=127.0.0.1
IP Address=0000:0000:0000:0000:0000:0000:0000:0001
DNS Name=172.24.199.75
IP Address=172.24.199.75
DNS Name=10.0.0.75
IP Address=10.0.0.75
```

7.2.2 Check certificates installed in SafeKit

You can use the `checkcert` command for checking all the certificates.

On each SafeKit nodes:

1. Log as administrator/root and open a command shell window
2. Change directory to `SAFE/web/bin`
3. Run `checkcert -t all`

   It checks all installed certificates and returns a failure if an error is detected
4. You can check that the server certificate contains some DNS name or IP address with:

   ```
   checkcert -h ”DNS name value”
   ```

   ```
   checkcert -i ”Numeric IP address value”
   ```

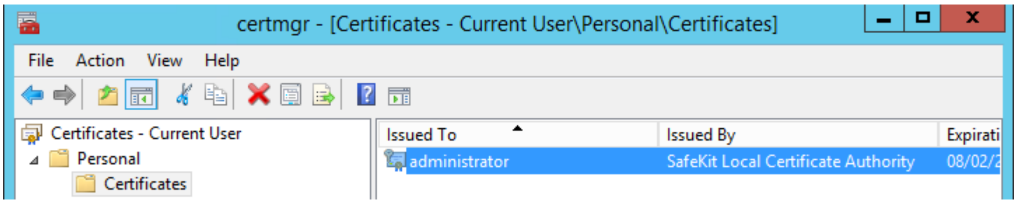
The server certificate must contain all DNS names and/or IP addresses used for HTTPS connection. These ones must also be included into the SafeKit cluster configuration file.

7.2.3 Check client certificates

When client certification authentication is configured, Certificate Authority and client certificates for the console must have been imported into the certificate store of the user’s workstation. Check that certificates are present into the expected store. Below is the procedure in Windows:

1. Log on to the workstation from which the user launches the console
2. Open a PowerShell console
3. Run `certmgr`
4. Locate Certificates - Current User\Personal\Certificates

   It must contain the client certificate for the web console

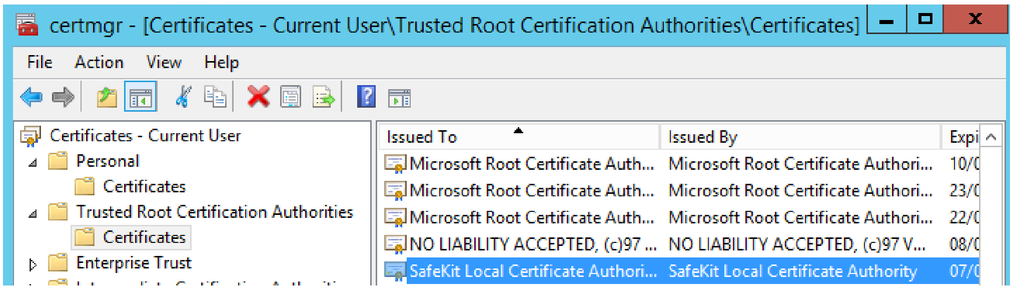


certmgr - [Certificates - Current User\Personal\Certificates]

Issued To	Issued By	Expirati
administrator	SafeKit Local Certificate Authority	08/02/2

   If the certificate is not in the proper store, remove it from the store and import it again as described in section 11.4.3.4 page 201.
5. Locate Certificates - Current User\Trusted Root Certification Authorities\Certificates

   It must contain the certificate of the Certification Authority used to generate the client certificate



certmgr - [Certificates - Current User\Trusted Root Certification Authorities\Certificates]

Issued To	Issued By	Expi
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	10/0
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	23/0
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	22/0
NO LIABILITY ACCEPTED, (c)97 ...	NO LIABILITY ACCEPTED, (c)97 V...	08/0
SafeKit Local Certificate Authori...	SafeKit Local Certificate Authority	07/0

If the certificate is not in the proper store, remove it from the store and import it again as described in section 11.4.3.5 page 202.

You must also clear the browser cache as described in 7.1.2 page 110.

7.2.4 Revert to HTTP configuration

If the problem can not be solved, you can revert to the HTTP configuration (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

httpd.webconsolessl.conf	On S1 and S2: ⇨ remove the file `SAFE/web/conf/ssl/httpd.webconsolessl.conf`
	On S1 and S2: ⇨ run `safekit webserver restart`

You must then clear the browser cache as described in 7.1.2 page 110.

7.3 How to read logs of the module?

SafeKit log for the module may be displayed using (replace below AM by the module name):

the web console/ Control/Select the node/Module Log tab/

the web console/ Configuration or Monitoring/ ▼ on the node/Support/Save log/

the command `safekit logview -m AM`

With the module log, you can understand why the module is no longer in its stable state ✓ (green) and ✓ (green).

Do not forget to also check output messages of **Application log** in the web console/ Control /Select the node /Application Log tab/ or in `SAFEVAR/modules/AM/userlog.ulog`

Note that a module can leave its stable state ✓ (green) and ✓ (green) because of an administrator command: `safekit start | stop | restart | swap | stopstart | forcestop… -m AM`

You will find a list of SafeKit log messages in the index: see Log Messages Index page 337.

Messages in the log after an administrator command are:

```
"Action start called by web@<IP>/SYSTEM/root"
"Action stop called by web@<IP>/SYSTEM/root"
"Action restart called by web@<IP>/SYSTEM/root"
"Action swap called by web@<IP>/SYSTEM/root"
"Action stopstart called by web@<IP>/SYSTEM/root"
"Action forcestop called by web@<IP>/SYSTEM/root"
```

web@<ip>: via the SafeKit console
SYSTEM: command on Windows
root: command on Linux

If "Stopping loop" appears in SafeKit log, see section 7.11 page 119

7.4 How to read the commands log of the server?

There is a log of the `safekit` commands ran on the server.

SafeKit commands log may be displayed using:

- the web console/ Control/Select the node/Commands Log tab/ (it displays `safekit` commands applied on the selected module and all global commands)
- the web console/ Advanced Configuration/Node tab/ Commands log/ (it displays all the commands logged on this server)
- on the server side, the command `safekit cmdlog`

  See section 10.9 page 175 for more details.

7.5 Stable module ✓ (green) and ✓ (green)

A stable mirror module on 2 servers is in the state ✓ `PRIM` (green) - ✓ `SECOND` (green): the application is running on the `PRIM` server; on failure, the `SECOND` server is ready to resume the application.

A stable farm module is in the state ✓ `UP` (green) on all servers of the farm: the application is running on all servers.

7.6 Degraded module ✓ (green) and ✗ (red)

A degraded mirror module is in the state ✓ `ALONE` (green) - ✗ `STOP`/`WAIT` (red). There is no recovery server, but the application is running on the `ALONE` server.

A degraded farm module is in the state ✓ `UP` (green) on at least one server of the farm, the other servers being in the state ✗ `STOP`/`WAIT` (red). The application is running on the `UP` server.

In the degraded case, there is no emergency procedure to implement. Analysis of the state ✗ `STOP`/`WAIT` (red) can be done later. However, you can attempt to restart the module in a stable state:

see 7.8 "Module ✗ `STOP` (red): restart the module" page 116

see 7.9 "Module ✗ `WAIT` (red): repair the resource="down"" page 117

7.7 Out of service module ✗ (red) and ✗ (red)

An out of service mirror or farm module is in the state ✗ `STOP`/`WAIT` (red) on all servers. In this case, the application is not operational on any server anymore. You must restore the situation and restart the module in ✓ (green) on at least one server:

see 7.8 “Module ✗ `STOP` (red): restart the module” page 116

see 7.9 “Module ✗ `WAIT` (red): repair the resource="down"” page 117

7.8 Module ✗ STOP (red): restart the module

To restart the stopped module (replace below AM by the module name):

web console/✓ Control / ▾ on the node/▶ Start/

or command `safekit start -m AM`

check that the module becomes ✓ (green)

And see results of start in the module and application logs:

web console/✓ Control/Select the node/Module Log tab/ and Application Log tab

or with the command `safekit logview -m AM` and `SAFEVAR/modules/AM/userlog.ulog`).

7.9 Module ✗ WAIT (red): repair the resource="down"

If the module is in the state ✗ WAIT (red), it waits for the state of a resource to become "up".

You must identify and fix the problem that caused the resource state to go down.

To determine the resource involved, see the log messages:

use web console/✔ Control/Select the node/Module Log tab/ or web console/✔ Control/Select the node/Resources tab/

or run the command `safekit logview -m AM` (replace AM by the module name)

Notes:

A wait checker is started after the prestart script and stopped before poststop

The checker is active on all servers ✓ ALONE/PRIM/SECOND/UP (green)

The action of the checker upon detecting an error is to set a resource to down

A failover rule referencing the resource performs the stopwait action

The module is locally in state ✗ WAIT (red) while the resource stays down

The module exits the ✗ WAIT (red) state as soon as the checker sets the resource back to up

Messages from wait checkers:

files not up-to-date locally: see section 5 page 95

"Data may be not uptodate for replicated directories (wait for the start of the remote server)"
"Action wait from failover rule notuptodate_server"
"If you are sure that this server has valid data, run safekit prim to force start as primary"

<interface check="on"> checker of a local network interface

"Resource intf.ip.0 set to down by intfcheck"
"Action wait from failover rule interface_failure"

<ping> checker of an external IP

"Resource ping.id set to down by pingcheck"
"Action wait from failover rule ping_failure"

<module>: checker of another module

"Resource module.othermodule_ip set to down by modulecheck"
"Action wait from failover rule module_failure"

<tcp ident="id" when="pre">: checker of an external TCP service

"Resource tcp.id set to down by tcpcheck"
"Action wait from failover rule tcpid_failure"

<custom ident="id" when="pre"> customized checker

"Resource custom.id set to down by customscript"
"Action wait from failover rule customid_failure"

<splitbrain> checker

“Resource splitbrain.uptodate set to down by splitbraincheck"

…

"Action wait from failover rule splitbrain_failure"

Files not up-to-date locally due to split-brain: see section 13.17 page 287

7.10 Module oscillating from ✓ (green) to ✓ (magenta)

If a module oscillates from state ✓ (green) to state ✓ (magenta), it is probably a victim of a restart or stopstart checker which detects a constant error.
By default, after the 4th unsuccessful restart on a server, the module stops, and the server stabilizes in ✗ `STOP` (red).
Use the SafeKit log to determine which checker is the source of the oscillation:
use web console/⚙ Control/Select the node/Module Log tab/
or run the command `safekit logview -m AM` (replace AM by the module name)
Notes:
A restart or stopstart checker is defined in `userconfig.xml` by when="prim"|"both" (mirror|farm)
`when="prim"`: checker started on the server ✓ `PRIM`/`ALONE` (green) after script `start_prim` (stopped before stop_prim) and checking the application started in `start_prim`
`when="both"`: checker started on all servers ✓ `UP` (green) after script `start_both` (stopped before `stop_both`) and checking the application started in `start_both`
The action of a checker on an error is to restart or stopstart the module. stopstart on ✓ `PRIM` (green) leads to a failover of the primary on the other server
The module is in the state ✓ `PRIM`/`UP` (magenta) during the application restart
After several oscillations, the modules stop with "Stopping loop" in SafeKit log: see section 7.11 page 119
	Messages from restart or stopstart checkers:
	<errd> in `userconfig.xml`: checker of processes
	"event atleast on proc appli.exe"
	"Action restart|stopstart called by errd"
	<tcp ident="id" when="prim"|"both"> in `userconfig.xml`: TCP checker of the application
	"Resource tcp.id set to down by tcpcheck"
	"Action restart|stopstart from failover rule tcp_failure"
	<custom ident="id" when="prim"|"both"> in `userconfig.xml`: custom checker
	"Resource custom.id set to down by customscript"
	"Action restart|stopstart from failover rule customid_failure"
	or
	"Action restart|stopstart called by customscript"

7.11 Message on stop after maxloop

If an error detected by a checker repeats itself several times and successively, the module is stopped on the server in ✗ `STOP` (red): because the error is permanent, and the action of the checker cannot correct it If in `userconfig.xml`, there is no parameter `maxloop` / `loop_interval` in <service>: by default, `maxloop="3" loop_interval="24"` if the checkers generate more than 3 unsuccessful restarts (restart, stopstart, stopwait) in less than 24H, then stop of module: ✗ `STOP` (red) The counter is reset to 0 if an administrator executes an action on the module such as `safekit start -m AM` (replace AM by the module name) or `safekit stop -m AM` (without the option `-i <identity>`)	Message on stop after maxloop "Stopping loop"

7.12 Module ✓ (green) but non-operational application

If a server has a status of ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green), the application can be non-operational because of undetected errors on start-up. Replace below AM by the module name.

Check the output messages of application scripts coming from `start_prim`/`start_both` and `stop_prim/stop_both`: they are visible in the web console/ Control/Select the node /Application Log tab/ or in `SAFEVAR/modules/AM/userlog.ulog`

check in Application log if there are errors during start or stop of the application. Be careful, sometimes the userlog is disabled because it is too large with <user logging="none"> in `userconfig.xml` of the module

check application scripts `start_prim`(/both) and stop_prim(/both) of a mirror(/farm) and `userconfig.xml`: they are visible in the web console/ Advanced Configuration / Installed modules/ module/ (bin and conf) or under `SAFE/modules/AM`

In case of a non-operational application, execute a restart on the module ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green) to stop and restart locally the application (without failover):

on ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green), run the command restart with the web console/ Control / ▼ on the node/Restart/ or with the command `safekit restart -m AM`

check that the application is operational on ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green)

If this procedure does not work, apply a stopstart of the module ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green) to stop and restart globally the module and the application (stopstart makes a failover to `SECOND` when `SECOND` is started):

on ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green), run the command stopstart with the web console/ Control/ ▼ on the node/Expert/StopStart/ or with the command `safekit stopstart -m AM`

check that the application is operational on ✓ `PRIM` (green) or ✓ `ALONE` (green) or ✓ `UP` (green)

7.13 Mirror module ✓ ALONE (green) / ✗ WAIT or STOP (red)

If a mirror module stays in state ✓ (green) ALONE / ✗ (red) WAIT, check the "remote state" resource on each node (Visible in web console / Control /Resources/Remote state). If this state is UNKNOWN on the two nodes, there is probably a communication problem between the nodes. This problem may also lead to ✓ (green) ALONE / ✗ (red) STOP.

Possible root causes are:

- Real network problem

  Check your network configurations on the two nodes.

- Firewall rules on one or the two nodes

  For details, see section 10.3 page 156

- Not the same SafeKit cluster configuration or cluster cryptographic keys

  To communicate, cluster nodes must belong to the same cluster and have the same configuration (see section 12 page 227):

  - The web console warns if nodes in the cluster nodes list have not an identical configuration
  - The command: `safekit cluster confinfo` on any nodes of the cluster must report an identical configuration signature for all nodes of the cluster (see 9.3 page 144)

  If the cluster configuration is not identical, re-apply the cluster configuration on all cluster nodes. (web console/ Advanced Configuration or Configuration /Cluster Configuration/select Advanced edit mode /Apply button)

- Not the same module cryptographic keys

  If cryptographic has been enabled for the module (the encryption resource is "on" in web console/ Control/Select the node/Resources tab) and the nodes do not have the same keys for the module, the nodes will not be able to communicate for the internal module communications. To distribute the same module cryptographic keys, re-apply the module configuration on all nodes (web console / Configuration/ ▼ on the module / Edit the configuration/Apply the Configuration/Apply button). See section 10.5 page 162 for details.

- Expired cryptographic keys

  In SafeKit <= 7.4.0.31, the key for encrypting the module communication has a validity period of 1 year. When it expires in a mirror module with file replication, the secondary fails to reintegrate and the module stops with an error message into the log:

  ```
  reintegre | D | XXX clnttcp_create: socket=7 TLS handshake failed
  ```

  In SafeKit > 7.4.0.31, the message is:

  ```
  reintegre | D | XXX clnttcp_create: socket=7 TLS handshake failed.
  Check server time and module certificate (expiration date, hash)
  ```

  To solve this problem, see 10.5.3.1 page 165

7.14 Farm module ✓ `UP` (green) but problem of load balancing in a farm

Even though all servers in the farm are ✓ `UP` (green), load balancing is not working.

7.14.1 Reported network load share are not coherent

In a farm module, the sum of the network load share of all ✓ `UP` (green), module nodes must be equal to 100% (See web console/ Control/Select the node/Resources tab/Network Load Share).

If it’s not the case, there is probably a communication problem between module nodes. Possible root causes are the same as for a mirror module. See section 7.13 page 121 for possible solutions.

See also section 4.3.6 page 83

7.14.2 virtual IP address does not respond properly

If the virtual IP does not respond properly to all requests for connections:

choose a server in the farm that receives and processes connections on the virtual IP address (established TCP connections): use the command (Windows) netstat –an | findstr <virtual IP address> or (Linux) netstat –an | grep <virtual IP address>

stop the farm module on all servers except the one that receives connections and that remains ✓ `UP` (green)

- either in the web console/ Control/▾ on the node / Stop/
- or run the command `safekit stop -m AM` (replace AM by the module name)

check that all connections to the virtual IP address are handled by the single server ✓ `UP` (green)

For a more detailed analysis on this topic, see:

4.3.4 “Test virtual IP address of a farm module” page 80

4.3.5 “Test TCP load balancing on a virtual IP address” page 82

4.3.7 “Test compatibility of the network with invisible MAC address” page 84

7.15 Problem after Boot

If you encounter a problem after boot, see section 4.1 page 69.

Note that by default, modules are not automatically started at boot. For this, you must setup the boot start into the module’s configuration:

⇨ with Edit the configuration that launches the web console/Configuration wizard (described in section 3.3.2 page 44)

or

⇨ in file `userconfig.xml` with the `boot` attribute of the `service` tag (see section 13.2.3 page 237)

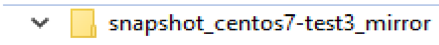
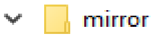
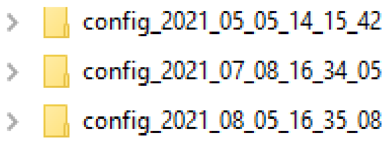
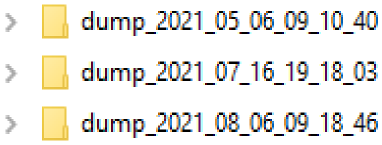
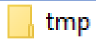
7.16 Analysis from snapshots of the module

When the problem is not easily identifiable, it is recommended to take a snapshot of the module on all nodes as described in section 3.5 page 54. A snapshot is a zip file that collects, for one module, the configuration files, dumps, ... Its content allows an offline and in-depth analysis of the module and node status.

Note

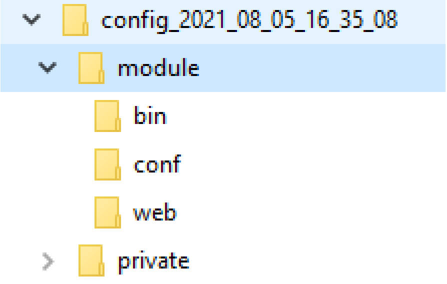
The structure and content of the snapshot varies depending on the version of SafeKit.

Since SafeKit 7.5, the structure of the snapshot is as follows:

 snapshot_centos7-test3_mirror	⇨ `snapshot_nodename_AM` snapshot for the module `AM` get from the node named `nodename`
 mirror	⇨ `AM` Application module name
 config_2021_05_05_14_15_42 config_2021_07_08_16_34_05 config_2021_08_05_16_35_08	⇨ `config_year_month_day_hour_mn_sec` Last 3 configurations for the module, including the current one
 dump_2021_05_06_09_10_40 dump_2021_07_16_19_18_03 dump_2021_08_06_09_18_46	⇨ `dump_year_month_day_hour_mn_sec` Last 3 dumps for the module, including the last one
 tmp	⇨ for the level 3 support

7.16.1 Module configuration files

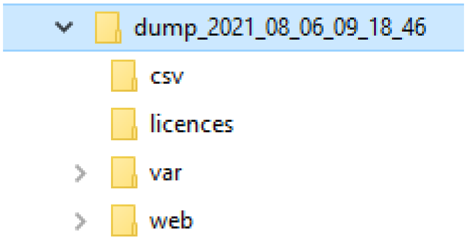
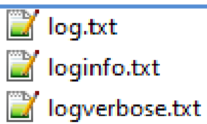
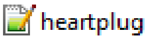
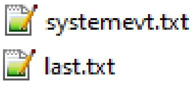
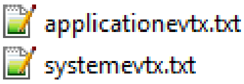
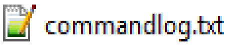
The module configuration files are saved as follows:

 config_2021_08_05_16_35_08 module bin conf web private	`module` directory contains the user configuration files ⇨ `bin` directory scripts `start_xx`, `stop_xx`, … ⇨ `conf` directory XML configuration `userconfig.xml`

⇨ Check the user configuration file and scripts for troubleshooting with the application integration into SafeKit

7.16.2 Module dump files

The dump contains the state of the module and the SafeKit node as it was at the time of the dump.

 dump_2021_08_06_09_18_46 csv licences var web	⇨ `csv` directory logs and status in csv format ⇨ `licences` directory SafeKit licenses get from `SAFE/conf` directory ⇨ `var` directory Extract of the `SAFEVAR` directory ⇨ `web` directory web server configuration get from `SAFE/web/conf` directory
 log.txt loginfo.txt logverbose.txt	⇨ Module logs (not verbose and verbose)
userlog.ulog	⇨ Application log
 heartplug	⇨ Information file Various information about the node (list and status of installed modules, OS version, disk, and network configuration, ...)
 systemevt.txt last.txt Or  applicationevtx.txt systemevtx.txt	⇨ System logs `last.txt` and `systemevt.txt` in Linux Or `applicationevt.txt` and `systemevt.txt` in Windows
 commandlog.txt	⇨ Commands log for the node
heart heart.trc nfsbox nfsbox.trc	⇨ Trace files for level 3 support

⇨ Check the license file(s) into `licenses` directory for troubleshooting with the SafeKit license check

⇨ Check the Apache configuration files into `web` directory for troubleshooting with the SafeKit web service

- Check the module logs, in `log.txt` and `logverbose.txt`, for troubleshooting with the module behavior
- Check the user scripts log `userlog.ulog` for troubleshooting with application start/stop
- If necessary, look at `heartplug` file for some information on the node and search the system logs for events that occurred at the same time as the problem being analyzed
- Check the commands log `commandlog.txt` for troubleshooting with cluster management or distributed commands

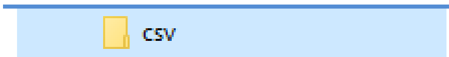
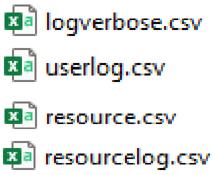
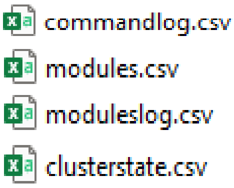
7.16.2.1 `var` directory

The `var` directory is mainly for the level 3 support. It is a copy of some part of the `SAFEVAR` directory. In the `var/cluster` directory:

- look at the `cluster.xml` file for checking the cluster configuration
- look at the `cluster_ip.xml` file for checking the DNS name resolution of names into the cluster configuration

7.16.2.2 `csv` directory

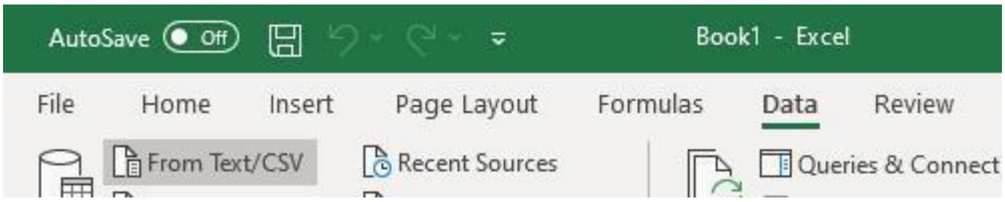
Since SafeKit 7.5, the logs and reports are also exported into csv format in the `csv` directory:

 csv	
 logverbose.csv userlog.csv resource.csv resourcelog.csv	⇨ Logs and status of the module Verbose log Application log Resources status Resources status history
 commandlog.csv modules.csv moduleslog.csv clusterstate.csv	⇨ Logs and status of the node Commands log List of installed modules For the level 3 support For the level 3 support

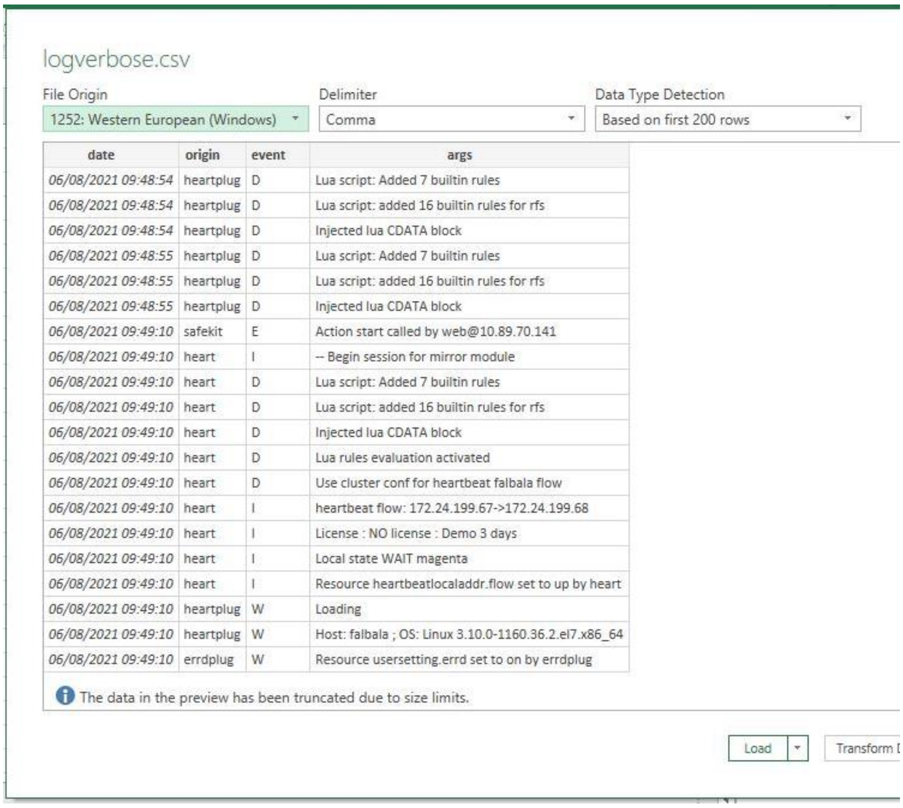
- Import the csv files into an Excel sheet to facilitate their analysis

  To import a file:

  1. Create a new sheet
  2. From the Data tab, import From Text/CSV



3. In the dialog box, locate and double-click the csv file to import, then click Import
4. Then click on Load


logverbose.csv

File Origin: 1252: Western European (Windows)

Delimiter: Comma

Data Type Detection: Based on first 200 rows

date	origin	event	args
06/08/2021 09:48:54	heartplug	D	Lua script: Added 7 builtin rules
06/08/2021 09:48:54	heartplug	D	Lua script: added 16 builtin rules for rfs
06/08/2021 09:48:54	heartplug	D	Injected lua CDATA block
06/08/2021 09:48:55	heartplug	D	Lua script: Added 7 builtin rules
06/08/2021 09:48:55	heartplug	D	Lua script: added 16 builtin rules for rfs
06/08/2021 09:48:55	heartplug	D	Injected lua CDATA block
06/08/2021 09:49:10	safekit	E	Action start called by web@10.89.70.141
06/08/2021 09:49:10	heart	I	-- Begin session for mirror module
06/08/2021 09:49:10	heart	D	Lua script: Added 7 builtin rules
06/08/2021 09:49:10	heart	D	Lua script: added 16 builtin rules for rfs
06/08/2021 09:49:10	heart	D	Injected lua CDATA block
06/08/2021 09:49:10	heart	D	Lua rules evaluation activated
06/08/2021 09:49:10	heart	D	Use cluster conf for heartbeat falbala flow
06/08/2021 09:49:10	heart	I	heartbeat flow: 172.24.199.67->172.24.199.68
06/08/2021 09:49:10	heart	I	License : NO license : Demo 3 days
06/08/2021 09:49:10	heart	I	Local state WAIT magenta
06/08/2021 09:49:10	heart	I	Resource heartbeatlocaladdr.flow set to up by heart
06/08/2021 09:49:10	heartplug	W	Loading
06/08/2021 09:49:10	heartplug	W	Host: falbala ; OS: Linux 3.10.0-1160.36.2.el7.x86_64
06/08/2021 09:49:10	errdplug	W	Resource usersetting.errd set to on by errdplug

The data in the preview has been truncated due to size limits.

You can use the Excel features to filter rows according to the level of the messages, ... and load in different sheets the csv of each node.

Note

For the exact date, format cells with N`umber/Custom jj/mm/aaaa hh:mm:ss,000`

7.17 Problem with the size of SafeKit databases

Since SafeKit 7.5, SafeKit uses SQLite3 storage to save:

⇨ The log and the status of the node

  - ✓ `SAFEVAR/log.db` contains the commands log
  - ✓ `SAFEVAR/resource.db` contains the list of installed modules and its history

  These are referred to as node databases.

⇨ The log and the resources of the module

  - ✓ `SAFEUSERVAR/log.db` contains the module log
  - ✓ `SAFEUSERVAR/resource.db` contains the state of the module resources and its history

  These are referred to as module databases.

The size of the logs and histories increases as events occur on the SafeKit node and modules. Therefore, they should be purged regularly by deleting the oldest entries. This is automatically done thanks to a periodic job (task scheduler in Windows; crontab in Linux) that is controlled by the `safeadmin` service. The clean of the node databases is always active. The clean of the module databases is active only when the module is running. To check that the jobs are ready:

⇨ Job for cleaning node databases

  - ✓ In Windows, run `schtasks /QUERY /TN safelog_clean`
  - ✓ In Linux, run `crontab -u safekit -l`

    The output of this command must contain the `safelog_clean` entry

⇨ Job for cleaning `AM` module databases (where `AM` is the module name)

  - ✓ In Windows, run `schtasks /QUERY /TN safelog_AM`
  - ✓ In Linux, run `crontab -u safekit -l`

    The output of this command must contain the `safelog_clean_AM` entry

The clean-up is implemented by a script located into `SAFEBIN` (in Linux, `SAFEBIN=/opt/safekit/private/bin`; in Windows, `SAFE=C:\safekit\private\bin` - if `%SYSTEMDRIVE%`=C:):

`dbclean.ps1` in Windows and `dbclean.sh` in Linux	Clean the log and history in the node databases
`dbclean.ps1 AM` in Windows and `dbclean.sh AM` in Linux	Clean the log and history in the databases of the module named `AM`

If necessary, you can run this script outside the scheduled period to force the databases clean-up.

7.18 Problem for retrieving the certification authority certificate from your PKI

When using your PKI, you must provide the certificate (the chain of certificates for the root and intermediates Certification Authorities) of:

- the certification authority `CA` (`cacert.crt` file) used to issue server certificates
- the certification authority `CLCA` (`clcacert.crt` file) used to issue client certificates, when client certificates authentication is used

If you have trouble retrieving these files from your PKI, you can build them using the procedure described below.

7.18.1 Export CA or CLCA certificate(s) from public certificates

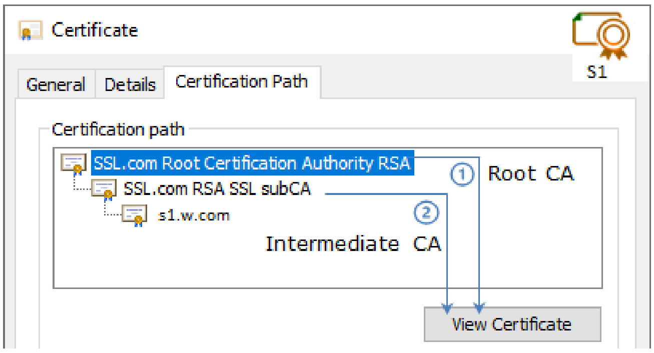
The following procedure explains how to build from a public certificate, the chain of certificates for the root and intermediates Certification Authorities, into the file `combined.cer`. This one can be used as:

- the `SAFE/web/conf/cacert.crt` file when it is generated from a server certificate
- the `SAFE/web/conf/clcacert.crt` file when it is generated from a client certificate. If different CLCAs are used to generate the different types of client certificates (distributed commands and web console certificates), run the following procedure for each client certificates. Then, concatenate each resulting `combined.cer` files into the final `clcacert.crt` file.

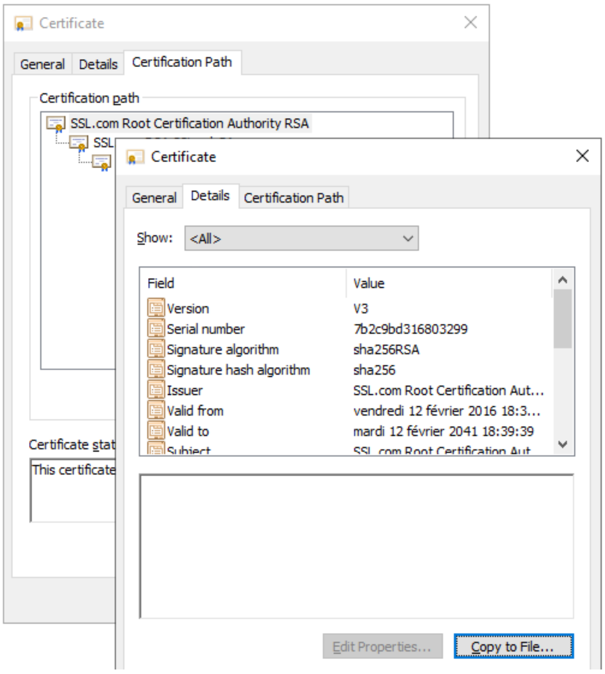
  When using a personal certificate for the web console, you may not have the associated public certificate. To get it, apply the procedure described in 7.18.2 page 130.

When you have the public certificate (.crt or .cer file in Base-64 encoded X.509 format) generated by your PKI:

1. Copy the .crt (or .cer) file on a Windows workstation
2. Double click on this file to open it with "Crypto Shell Extensions"
3. Select the "Certification Path" tab to view the tree of certification authorities
4. Select an entry (from top to down except the leaf)



5. Click on "View Certificate". A new window is opened with details for the selected certificate
6. In this new window, select the "Details" tab and click "Copy to File"

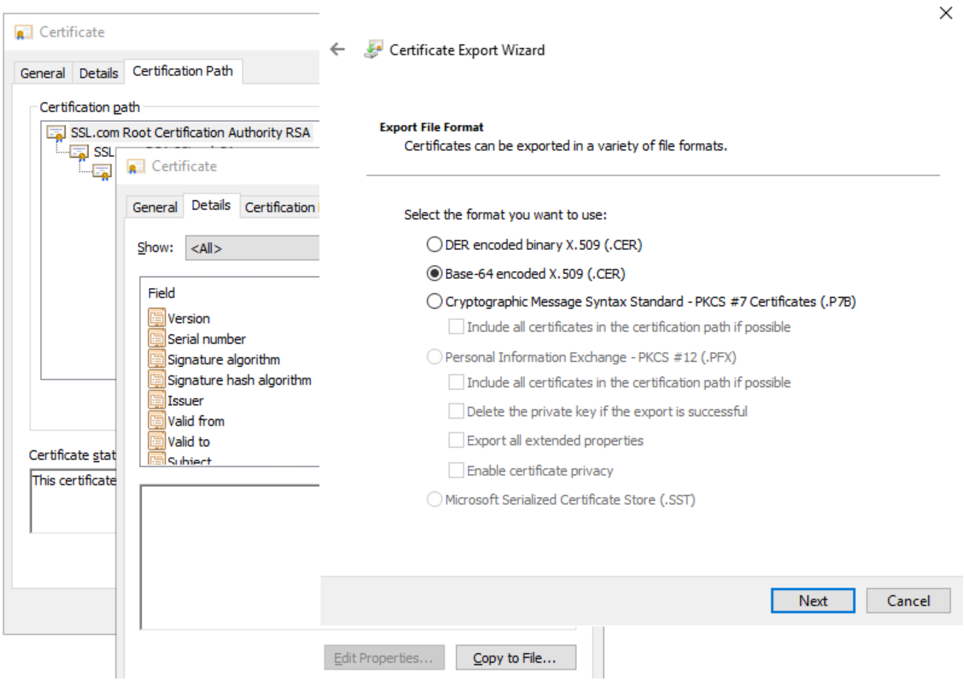


Field	Value
Version	V3
Serial number	7b2c9bd316803299
Signature algorithm	sha256RSA
Signature hash algorithm	sha256
Issuer	SSL.com Root Certification Aut...
Valid from	vendredi 12 février 2016 18:3...
Valid to	mardi 12 février 2041 18:39:39
Subject	SSL.com Root Certification Aut

7. It opens the Certificate Export Wizard:

   a. Click on “Next” to continue

   b. On the “Export File Format” page, select “Base-64 encoded X.509 (.CER).”, and then click “Next”



Export File Format
Certificates can be exported in a variety of file formats.

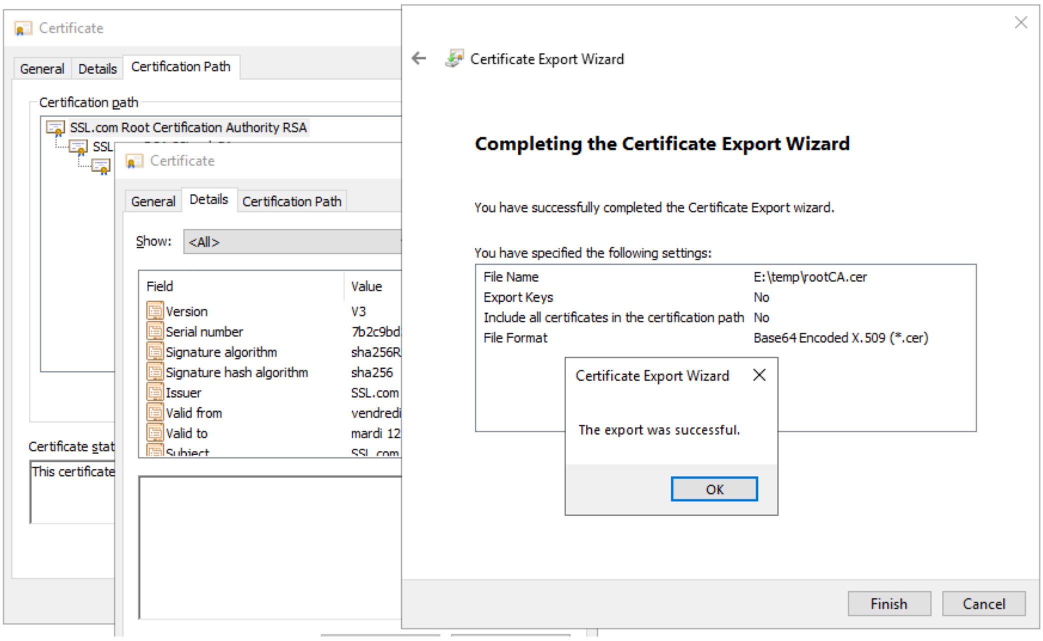
Select the format you want to use:

○ DER encoded binary X.509 (.CER)
● Base-64 encoded X.509 (.CER)
○ Cryptographic Message Syntax Standard - PKCS #7 Certificates (.P7B)
□ Include all certificates in the certification path if possible
○ Personal Information Exchange - PKCS #12 (.PFX)
□ Include all certificates in the certification path if possible
□ Delete the private key if the export is successful
□ Export all extended properties
□ Enable certificate privacy
○ Microsoft Serialized Certificate Store (.SST)

   c. For “File to Export”, “Browse” to the location to which you want to export the certificate. Fill “File name” with the name of the certificate file. Then, click “Next”

   d. Click “Finish” to export the certificate

   e. Your certificate is successfully exported



Completing the Certificate Export Wizard

You have successfully completed the Certificate Export wizard.

You have specified the following settings:

File Name	E:\temp\rootCA.cer
Export Keys	No
Include all certificates in the certification path	No
File Format	Base64 Encoded X.509 (*.cer)

Certificate Export Wizard

The export was successful.

8. Now repeat steps 4-7 for all entries (except the last one) to export all intermediate CA certificates in the Base-64 encoded X.509(.CER) format. For the example, you would repeat steps 4-7 on SSSL.com RSA subCA intermediate CA to extract it as its own certificate.
9. Concatenate all your CA certificates into one file `combined.cer`

   Run the following command with all the CA certificates you extracted earlier:

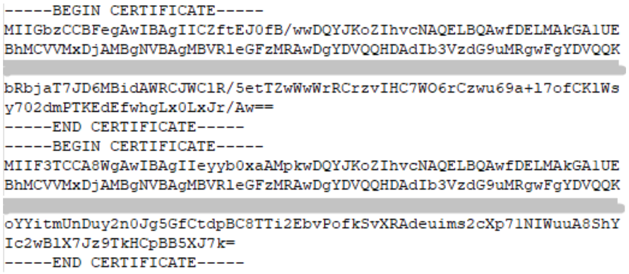
   ➾ In Windows:

   ```
   type intermediateCA.cer rootCA.cer > combined.cer
   ```

   ➾ In Linux:

   ```
   cat intermediateCA.cer rootCA.cer >> combined.cer
   ```

   The resulting combined certificate should look something like the following:



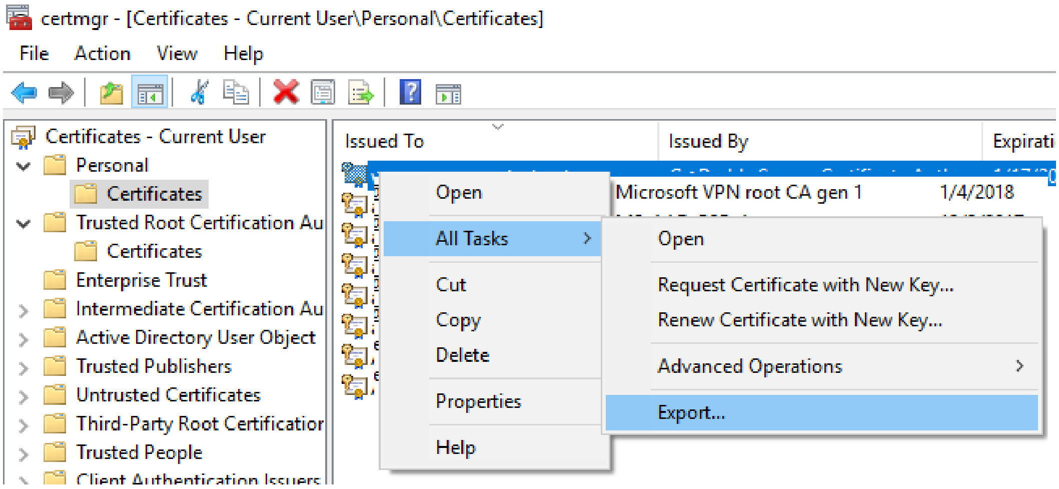
```
-----BEGIN CERTIFICATE-----
MIIGbzCCBFegAwIBAgIICZftEJ0fB/wwDQYJKoZIhvcNAQELBQAwfDELMAkGA1UE
BhMCVVMxDjAMBgNVBAgMBVRleGFzMRAwDgYDVQQHDAdIb3VzdG9uMRgwFgYDVQQK

bRbjaT7JD6MBidAWRCJWClR/5etTZwWwWrRCrzvIHC7WO6rCzwu69a+l7ofCKlWs
y702dmPTKEdEfwhgLx0LxJr/Aw==
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIF3TCCA8WgAwIBAgIIeyyb0xaAMpkwDQYJKoZIhvcNAQELBQAwfDELMAkGA1UE
BhMCVVMxDjAMBgNVBAgMBVRleGFzMRAwDgYDVQQHDAdIb3VzdG9uMRgwFgYDVQQK

oYYitmUnDuy2n0Jg5GfCtdpBC8TTi2EbvPofkSvXRAdeuims2cXp7lNIWuuA8ShY
Ic2wBlX7Jz9TkHCpBB5XJ7k=
-----END CERTIFICATE-----
```

7.18.2 Export public certificate

When using your personal certificate for the web console, you may not have the associated public certificate. To get it, apply the following procedure:

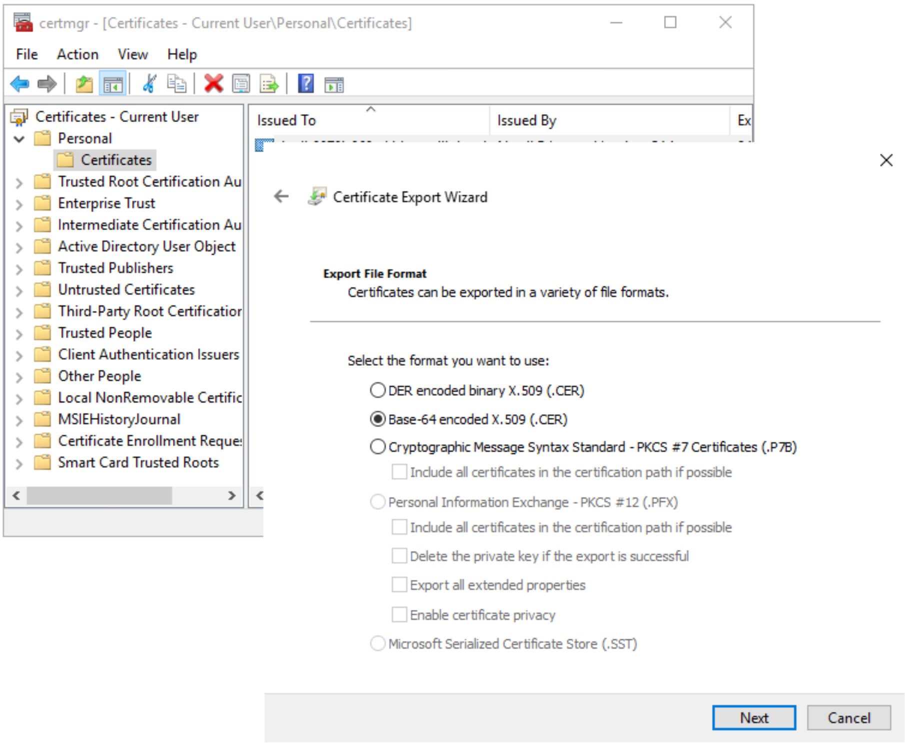
1. On your Windows workstation, open "Manage user certificates" (`certmgr.msc`)

2. Locate the certificate, typically in “Certificates - Current User\Personal\Certificates”, and right-click. If the user has several certificates, select the one with "Client Authentication" as "Expected Roles" and whose "Expiration Date" has not passed
3. Click “All Tasks”, and then click “Export”. This opens the Certificate Export Wizard.


certmgr - [Certificates - Current User\Personal\Certificates]
File Action View Help
Certificates - Current User
Personal
Certificates
Trusted Root Certification Au
Certificates
Enterprise Trust
Intermediate Certification Au
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification
Trusted People
Client Authentication Issuers
Issued To
Issued By
Expirati
Microsoft VPN root CA gen 1
1/4/2018
Open
All Tasks
Cut
Copy
Delete
Properties
Help
Open
Request Certificate with New Key...
Renew Certificate with New Key...
Advanced Operations
Export...

If you can't find the certificate under “Current User\Personal\Certificates”, you may have accidentally opened "Certificates - Local Computer", rather than "Certificates - Current User". If you want to open Certificate Manager in current user scope using PowerShell, you type `certmgr` in the console window.

4. In the Certificate Export Wizard, click “Next”
5. Select “No, do not export the private key”, and then click “Next”
6. On the “Export File Format” page, select “Base-64 encoded X.509 (.CER).”, and then click “Next”

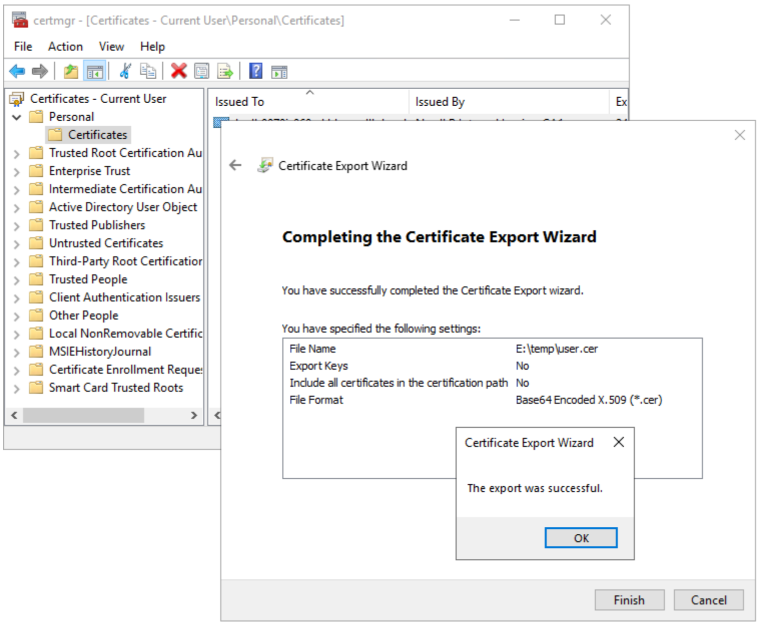

certmgr - [Certificates - Current User\Personal\Certificates]
File Action View Help
Certificates - Current User
Personal
Certificates
Trusted Root Certification Au
Enterprise Trust
Intermediate Certification Au
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certificatior
Trusted People
Client Authentication Issuers
Other People
Local NonRemovable Certific
MSIEHistoryJournal
Certificate Enrollment Reque:
Smart Card Trusted Roots
Issued To
Issued By
Ex
Certificate Export Wizard
Next
Cancel

Export File Format
Certificates can be exported in a variety of file formats.

Select the format you want to use:

○ DER encoded binary X.509 (.CER)
● Base-64 encoded X.509 (.CER)
○ Cryptographic Message Syntax Standard - PKCS #7 Certificates (.P7B)
□ Include all certificates in the certification path if possible
○ Personal Information Exchange - PKCS #12 (.PFX)
□ Include all certificates in the certification path if possible
□ Delete the private key if the export is successful
□ Export all extended properties
□ Enable certificate privacy
○ Microsoft Serialized Certificate Store (.SST)

7. For “File to Export”, “Browse” to the location to which you want to export the certificate. Fill “File name” with the name of the certificate file. Then, click “Next”
8. Click “Finish” to export the certificate
9. Your certificate is successfully exported


certmgr - [Certificates - Current User\Personal\Certificates]
File
Action
View
Help
Certificates - Current User
Personal
Certificates
Trusted Root Certification Au
Enterprise Trust
Intermediate Certification Au
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certificatior
Trusted People
Client Authentication Issuers
Other People
Local NonRemovable Certific
MSIEHistoryJournal
Certificate Enrollment Reque:
Smart Card Trusted Roots
Issued To
Issued By
Ex
Certificate Export Wizard
Finish
Cancel
OK

Completing the Certificate Export Wizard

You have successfully completed the Certificate Export wizard.

You have specified the following settings:

File Name	E:\temp\user.cer
Export Keys	No
Include all certificates in the certification path	No
File Format	Base64 Encoded X.509 (*.cer)

Certificate Export Wizard

The export was successful.

The exported certificate looks like this:

At this step, you can apply the procedure described in 7.18.1 page 128, to export the Certification Authority certificate(s) from this public certificate.

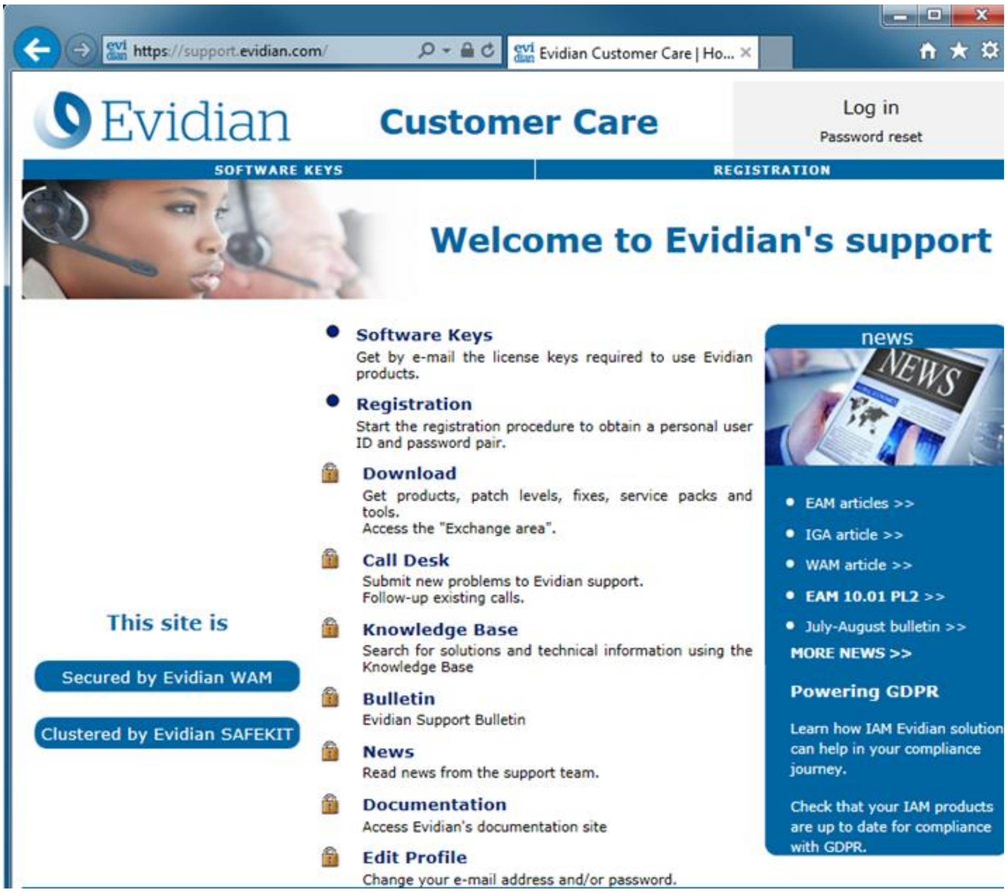
7.19 Still in Trouble

See Messages Index page 337

See section 8.5 page 135 for opening a ticket at the call desk

8. Access to Evidian support

8.1 Home page of support site



Welcome to Evidian's support

Software Keys
Get by e-mail the license keys required to use Evidian products.

Registration
Start the registration procedure to obtain a personal user ID and password pair.

Download
Get products, patch levels, fixes, service packs and tools.
Access the "Exchange area".

Call Desk
Submit new problems to Evidian support.
Follow-up existing calls.

Knowledge Base
Search for solutions and technical information using the Knowledge Base

Bulletin
Evidian Support Bulletin

News
Read news from the support team.

Documentation
Access Evidian's documentation site

Edit Profile
Change your e-mail address and/or password.

This site is

Secured by Evidian WAM

Clustered by Evidian SAFEKIT

news

- EAM articles >>
- IGA article >>
- WAM article >>
- EAM 10.01 PL2 >>
- July-August bulletin >>

MORE NEWS >>

Powering GDPR

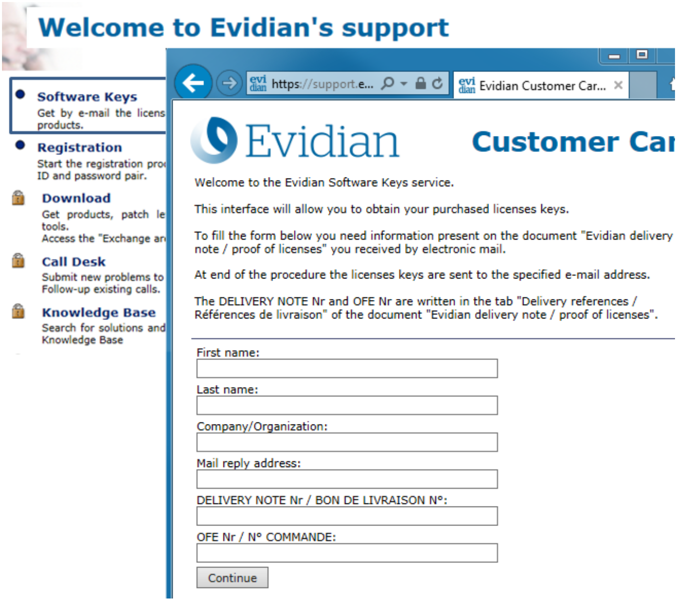
Learn how IAM Evidian solution can help in your compliance journey.

Check that your IAM products are up to date for compliance with GDPR.

- https://support.evidian.com
- Software Keys: get permanent keys
- Subscription Request: create an account
- Download: download product or upload snapshots
- Call desk: tool for opening a call on problem
- Knowledge Base: base of KB

8.2 Permanent license keys

- ➾ https://support.evidian.com
- ➾ Software Keys: get permanent keys
- ➾ Fill-in the form with the delivery note sent after a purchase order
- ➾ Take "hostname" and OS of your servers
- ➾ To obtain a temporary key for any hostname and any OS, for details see section 2.1.5 page 28



Welcome to Evidian's support

- Software Keys
  Get by e-mail the licens
  products.
- Registration
  Start the registration pro
  ID and password pair.
- Download
  Get products, patch le
  tools.
  Access the "Exchange ar
- Call Desk
  Submit new problems to
  Follow-up existing calls.
- Knowledge Base
  Search for solutions and
  Knowledge Base

Evidian Customer Ca

Welcome to the Evidian Software Keys service.

This interface will allow you to obtain your purchased licenses keys.

To fill the form below you need information present on the document "Evidian delivery note / proof of licenses" you received by electronic mail.

At end of the procedure the licenses keys are sent to the specified e-mail address.

The DELIVERY NOTE Nr and OFE Nr are written in the tab "Delivery references / Références de livraison" of the document "Evidian delivery note / proof of licenses".

First name:

Last name:

Company/Organization:

Mail reply address:

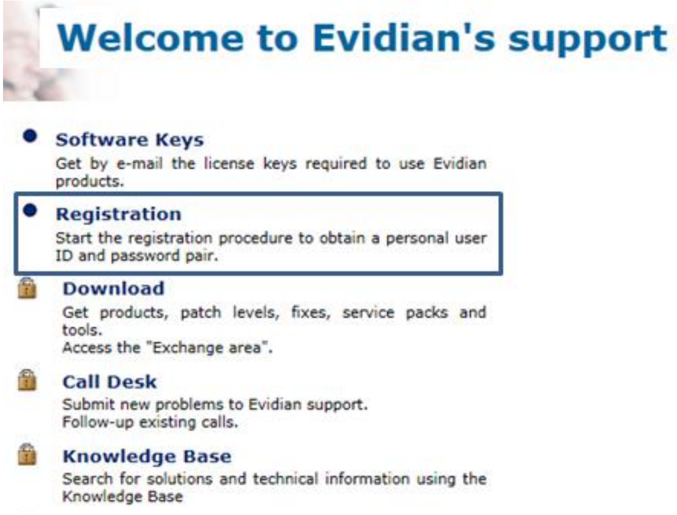
DELIVERY NOTE Nr / BON DE LIVRAISON N°:

OFE Nr / N° COMMANDE:

Continue

8.3 Create an account

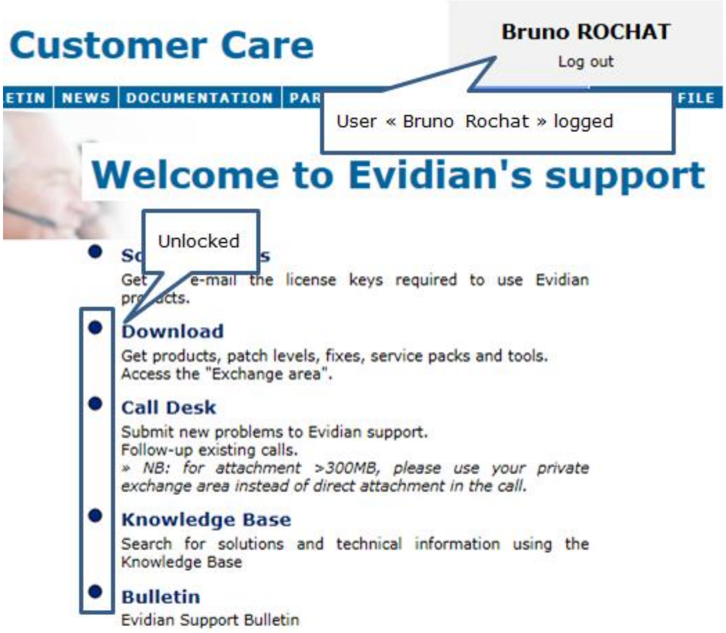
- ➾ https://support.evidian.com
- ➾ Subscription Request: create an account
- ➾ The procedure must be executed once with:
  - Your client identity
  - Your confidential identity
  - A unique e-mail address
- ➾ Note: your identities are sent by mail if you take an Evidian support contract
- ➾ What you will obtain: a user account and a private password on the site



Welcome to Evidian's support

- Software Keys
  Get by e-mail the license keys required to use Evidian products.
- Registration
  Start the registration procedure to obtain a personal user ID and password pair.
- Download
  Get products, patch levels, fixes, service packs and tools.
  Access the "Exchange area".
- Call Desk
  Submit new problems to Evidian support.
  Follow-up existing calls.
- Knowledge Base
  Search for solutions and technical information using the Knowledge Base

8.4 Access to your account



User « Bruno Rochat » logged

Welcome to Evidian's support

- **So s**
  Get e-mail the license keys required to use Evidian pr cts.
- **Download**
  Get products, patch levels, fixes, service packs and tools.
  Access the "Exchange area".
- **Call Desk**
  Submit new problems to Evidian support.
  Follow-up existing calls.
  » NB: for attachment >300MB, please use your private exchange area instead of direct attachment in the call.
- **Knowledge Base**
  Search for solutions and technical information using the Knowledge Base
- **Bulletin**
  Evidian Support Bulletin

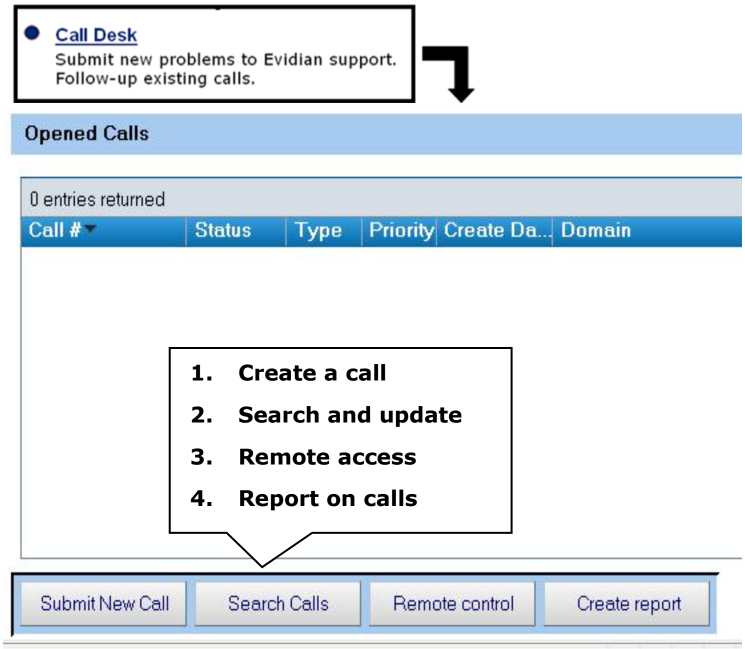
⇨ https://support.evidian.com

⇨ Login on top at right with your identity and password

⇨ Then you have access to all services of support site

8.5 Call desk to open a trouble ticket

8.5.1 Call desk operations



● Call Desk
Submit new problems to Evidian support.
Follow-up existing calls.

Opened Calls

0 entries returned

Call #	Status	Type	Priority	Create Da...	Domain

1. **Create a call**
2. **Search and update**
3. **Remote access**
4. **Report on calls**

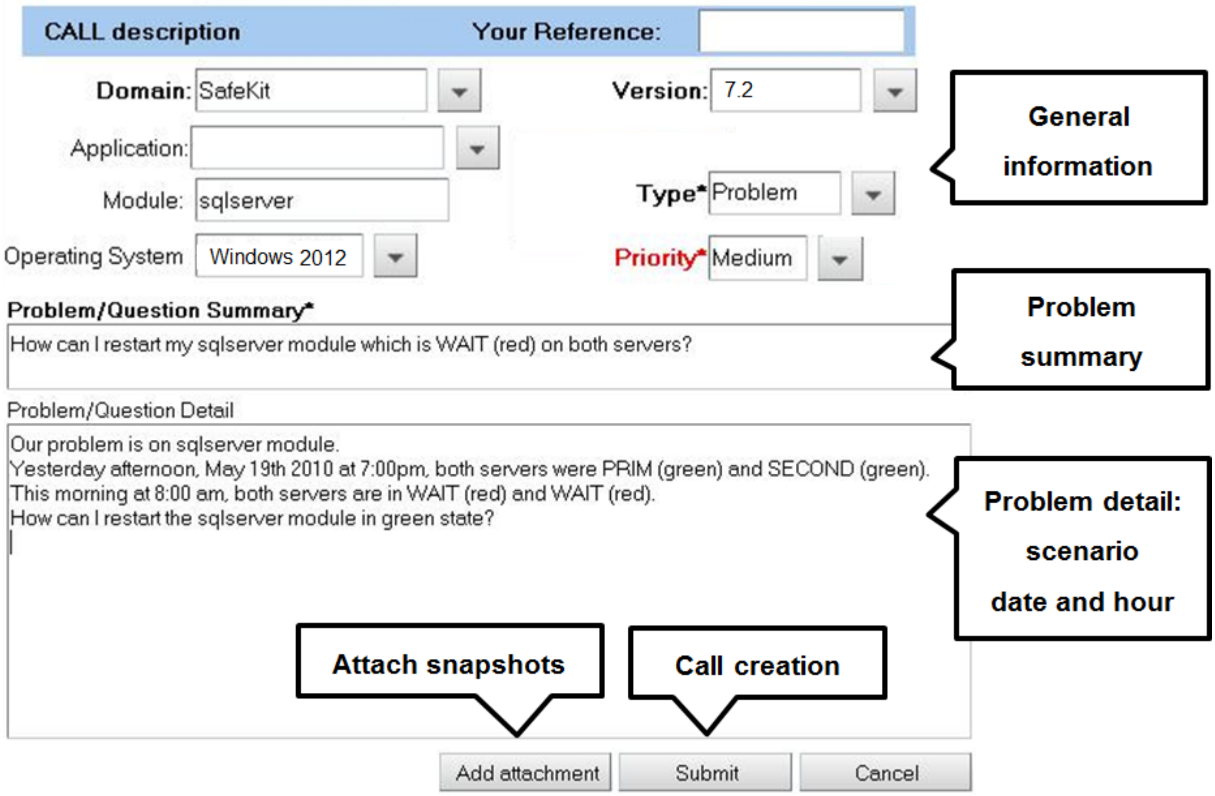
⇨ https://support.evidian.com

⇨ Call desk: tool to open a trouble ticket on problem with 2 main operations

⇨ Create a call

⇨ Search for a Call and exchange with support on a Call

8.5.2 Create a call



CALL description   Your Reference:

Domain: SafeKit   Version: 7.2

Application:

Module: sqlserver   Type* Problem

Operating System Windows 2012   Priority* Medium

Problem/Question Summary*

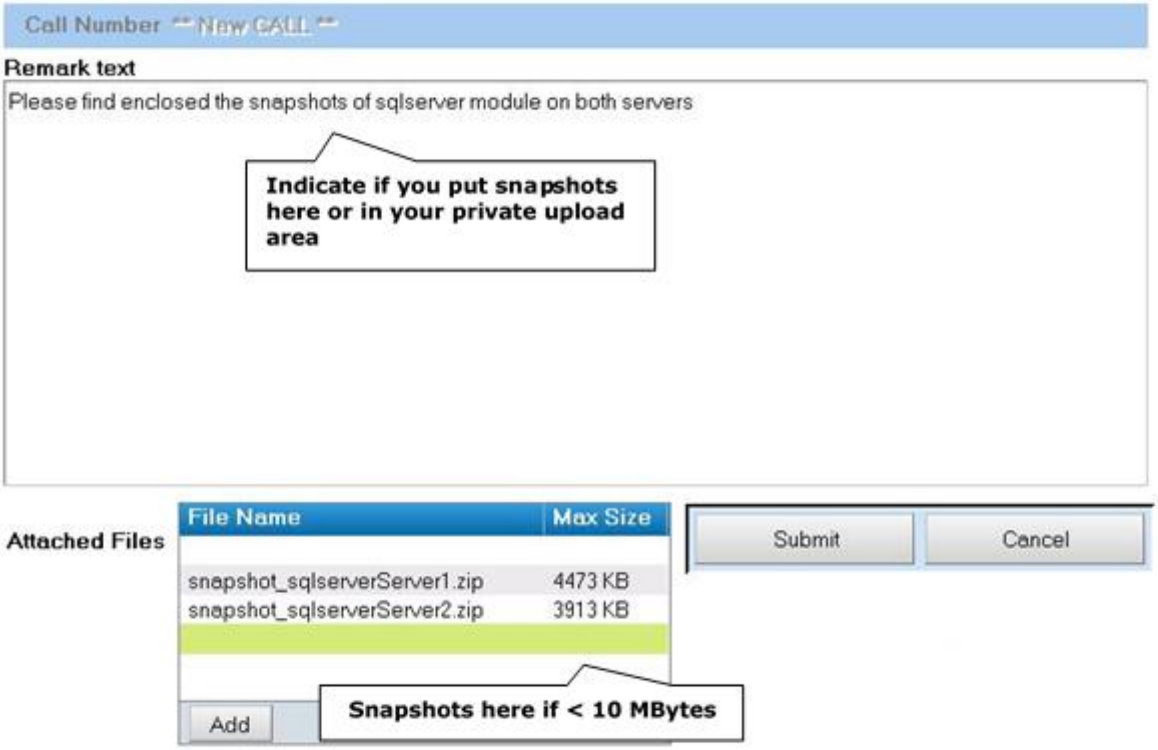
How can I restart my sqlserver module which is WAIT (red) on both servers?

Problem/Question Detail

Our problem is on sqlserver module.
Yesterday afternoon, May 19th 2010 at 7:00pm, both servers were PRIM (green) and SECOND (green).
This morning at 8:00 am, both servers are in WAIT (red) and WAIT (red).
How can I restart the sqlserver module in green state?

- ➾ In the header, specify the SafeKit version, problem type and priority as well as the module name and the OS
- ➾ Summarize the problem and then describe with more details the scenario and the date and time of the problem
- ➾ Snapshots of the SafeKit module causing problem are necessary for the analysis. See next section for attaching snapshots
- ➾ Create the call by pressing "Submit"

8.5.3 Attach the snapshots



Call Number ** New CALL **

Remark text

Please find enclosed the snapshots of sqlserver module on both servers

Indicate if you put snapshots here or in your private upload area

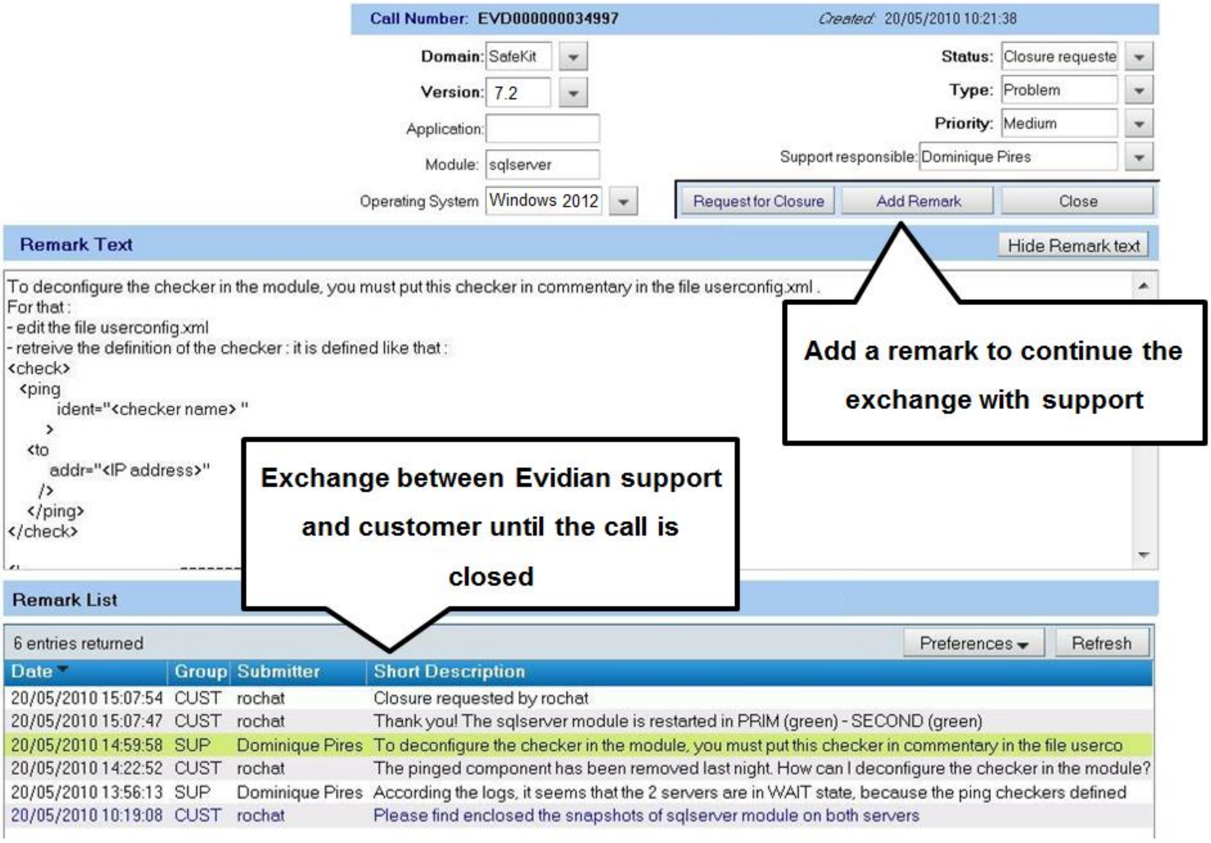
Attached Files

File Name	Max Size
snapshot_sqlserverServer1.zip	4473 KB
snapshot_sqlserverServer2.zip	3913 KB

Snapshots here if < 10 MBytes

- When there is a problem on a SafeKit module, snapshots of the module on all servers are necessary for analysis
- To get snapshots, see section 3.5 page 54
- If the snapshots size is smaller than 10 MBytes, you can attach them with the opening of the call by clicking on "Add"
- Otherwise, downloading snapshots on the support site may take several minutes. In this case indicate in "Remark text" that you download them into your private upload area: see section 8.6.3 page 140

8.5.4 Answers to a call and exchange with support



Call Number: EVD000000034997 Created: 20/05/2010 10:21:38

Domain: SafeKit

Version: 7.2

Application:

Module: sqlserver

Operating System Windows 2012

Status: Closure requeste

Type: Problem

Priority: Medium

Support responsible: Dominique Pires

```
To deconfigure the checker in the module, you must put this checker in commentary in the file userconfig.xml .
For that :
- edit the file userconfig.xml
- retreive the definition of the checker : it is defined like that :
<check>
  <ping
      ident="<checker name> "
      >
   <to
     addr="<IP address>"
    />
   </ping>
</check>
```

Add a remark to continue the exchange with support

Exchange between Evidian support and customer until the call is closed

6 entries returned

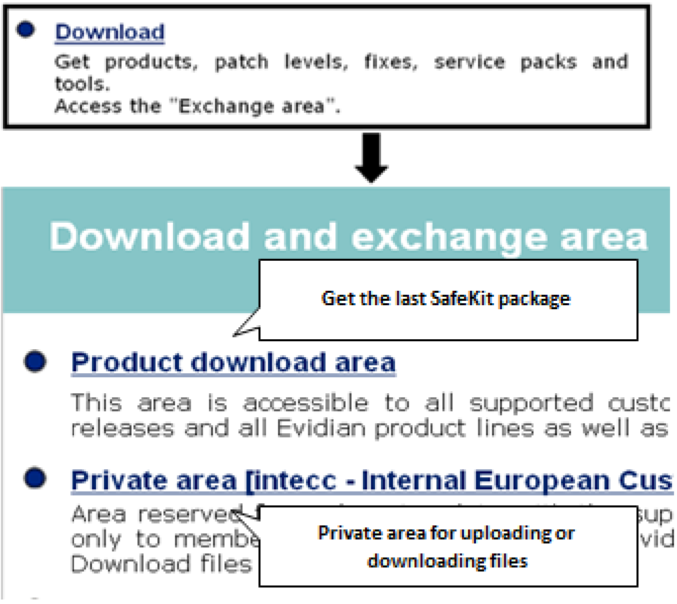
Date	Group	Submitter	Short Description
20/05/2010 15:07:54	CUST	rochat	Closure requested by rochat
20/05/2010 15:07:47	CUST	rochat	Thank you! The sqlserver module is restarted in PRIM (green) - SECOND (green)
20/05/2010 14:59:58	SUP	Dominique Pires	To deconfigure the checker in the module, you must put this checker in commentary in the file userco
20/05/2010 14:22:52	CUST	rochat	The pinged component has been removed last night. How can I deconfigure the checker in the module?
20/05/2010 13:56:13	SUP	Dominique Pires	According the logs, it seems that the 2 servers are in WAIT state, because the ping checkers defined
20/05/2010 10:19:08	CUST	rochat	Please find enclosed the snapshots of sqlserver module on both servers

- All exchanges between the support and the customer are made with "Remarks"
- When support adds a remark on a call, the customer is notified by mail. This is the case for first response of the support after the opening of the call
- After consultation of the last remark of support, the customer can add a new remark in turn
- The exchange takes place until the closure of the call by agreement between the customer and Evidian support

8.6 Download and upload area

8.6.1 Two areas of download and upload

- ➾ https://support.evidian.com
- ➾ Product download area: area for downloading SafeKit packages
- ➾ Private area [client identity]: private area to upload files



● Download
Get products, patch levels, fixes, service packs and tools.
Access the "Exchange area".

Download and exchange area

Get the last SafeKit package

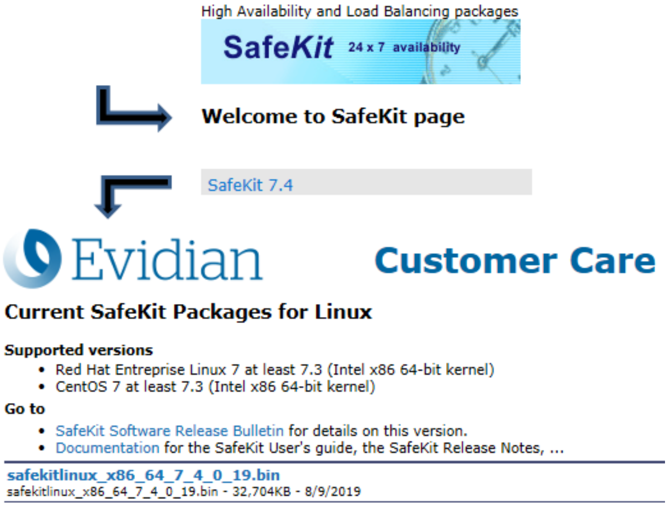
● Product download area
This area is accessible to all supported custc
releases and all Evidian product lines as well as

● Private area [intecc - Internal European Cus
Area reserved
only to membe
Download files

Private area for uploading or downloading files

8.6.2 Product download area

- ➾ Go to <Version 7.5>/Platforms/<Your platform>/Current versions
- ➾ Download the SafeKit package
- ➾ For more information on installation, documentation, upgrade, see section 2 page 25



High Availability and Load Balancing packages
SafeKit 24 x 7 availability

Welcome to SafeKit page

SafeKit 7.4

Evidian Customer Care

Current SafeKit Packages for Linux

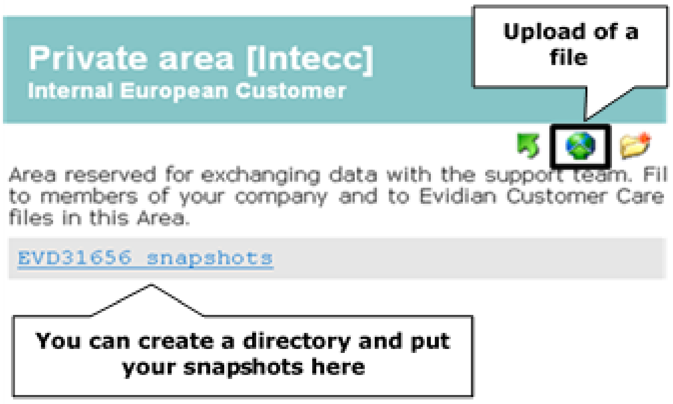
Supported versions
- Red Hat Entreprise Linux 7 at least 7.3 (Intel x86 64-bit kernel)
- CentOS 7 at least 7.3 (Intel x86 64-bit kernel)

Go to
- SafeKit Software Release Bulletin for details on this version.
- Documentation for the SafeKit User's guide, the SafeKit Release Notes, ...

safekitlinux_x86_64_7_4_0_19.bin
safekitlinux_x86_64_7_4_0_19.bin - 32,704KB - 8/9/2019

8.6.3 Private upload area

- Create a directory for a problem
- Upload snapshots in this directory with
- For building snapshots, see section 3.5 page 54
- For attaching snapshots, see section 8.5.3 page 137



Private area [Intecc]

Internal European Customer

Area reserved for exchanging data with the support team. Fil to members of your company and to Evidian Customer Care files in this Area.

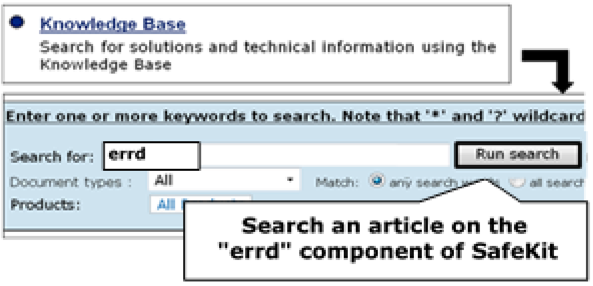
EVD31656 snapshots

Upload of a file

You can create a directory and put your snapshots here

8.7 Knowledge base

- https://support.evidian.com
- Knowledge Base: base of KB
- Search for example all articles on the errd component of SafeKit



● Knowledge Base

Search for solutions and technical information using the Knowledge Base

Enter one or more keywords to search. Note that '*' and '?' wildcard

Search for: errd

Document types : All

Match: ● any search w... ○ all search

Products: All

Search an article on the "errd" component of SafeKit

9.Command line interface

9.1 Distributed commands

Almost all `safekit` commands can be applied on a list of cluster nodes.
Exceptions are `safekit logview`, `safekit -p` and `safekit -r` commands which can be used only locally.
The distributed command line interface requires the execution of the SafeKit web service on each node of the list (see section 10.6 page 167).

`safekit -H <url> [,<url,...] <action> <arg>`	Execute action on servers specified by the URL list. URLs must be separated by commas.
	Instead of URLs, it is possible to use a comma separated list of server names as they appear in the cluster.xml file. Associated URLs are automatically built as `https:9453` or `http:9010` (depending on `SAFE/web/conf/ssl/` content)
	The special syntax –H "*" stands for all the nodes declared in the cluster.xml admin lan.
	To override protocol and port, use the [<protocol>:<port>] syntax. The ':<port>' part is optional. Protocol may be 'http' or 'https'. Default port for http protocol is 9010.
	Example: `safekit -H http://192.168.0.2:9010,http://192.168.0.3:9010 module list`
	`safekit -H "*" module list`
	`safekit -H "[http],*" module list`
	`safekit -H "[https:9500],server1,server2" module list`

`safekit`
`[-H <url>[,...]]`
`-E <module>`
	Deploy the locally installed <module> on the servers specified -H parameter.
	This command performs the following actions:
	creates <module>.safe from local `SAFE/modules/<module>`
	transfers and installs <module>.safe on the list of servers
	if the module was configured locally, configures it on remote servers
	Example: `safekit -E farm` will export the local farm module to the list of servers specified in `SAFEVAR/default_cluster.txt` (see example above for syntax of `default_cluster.txt`)

`safekit [-H <url>[,…] -G`	Deploy the local cluster configuration files on all the servers specified–H. This command performs the following actions: Collect the content of the `SAFEVAR/cluster` directory Transfer and copy the collected files into the target servers' `SAFEVAR/cluster` directory Trigger `safeadmin` configuration reload

9.2 Command lines for boot and for shutdown

Use the following commands for starting/stopping SafeKit services, configuring services and modules automatic start/stop on boot/shutdown, stopping all running modules.

In Windows, you may have to apply the procedure described in 10.4 page 161.

`safeadmin`
`(Windows)`
	SafeKit main service mandatory and started automatically at boot. `safeadmin` can be controlled using the Windows Services Control Panel applet

`service safeadmin start (Linux)`	SafeKit main service mandatory and started automatically at boot
`safekit webserver [start | stop | restart]`	Controls start/stop/restart of the `safewebserver` service. This service is used by the web console, module checkers and distributed command line interface. The command starts the httpd processes and waits for their start-up
`safekit safeagent [start | stop | restart | check]`	Controls start/stop of the `safeagent` service that implements the SafeKit SNMP agent

`safekit boot [webon | weboff | webstatus]`	Controls the automatic start at boot of the `safewebserver` service ("on" or "off"; by default, "on")
`safekit boot [snmpon | snmpoff | snmpstatus]`	Controls the automatic start at boot of the `safeagent` service ("on" or "off"; by default, "off")
`safekit boot [-m AM] [on | off | status]`	Controls whether the `AM` module starts automatically at boot or not ("on" or "off"; by default, "off") Without the option `-m AM`, lists the boot status of all modules. Important Since SafeKit 7.5, the boot start of a module can be defined in the module configuration with the `boot` attribute of the `service` tag in `userconfig.xml`. This configuration option makes the `safekit boot -m AM on | off` deprecated. However, this is still supported and replaces the module configuration, provided that the `boot` attribute is not present or set with the value `ignore`.
`safekit shutdown`	Stops all running modules

9.3 Command lines to configure and monitor safekit cluster

`safekit cluster config [filepath .xml or .zip] [lock | unlock]`	Apply the new SafeKit cluster configuration with the content of the file passed as argument, cluster.xml or cluster.zip: ⇨ cluster.xml configure with new cluster.xml and generate new cryptographic keys ⇨ cluster.zip configure with the new cluster.xml and cryptographic keys stored into the zip file When called with no argument, this command keeps the current configuration but generates new cryptographic keys. Ex: `safekit cluster config /tmp/newcluster.xml` Note Use with great care: the new cluster configuration and cryptographic key must then be copied to all cluster nodes to have the same cluster configuration on all nodes. If the command is called with the parameter `lock`, future `safekit cluster config` commands will not be granted until they are called with the `unlock` parameter.
`safekit cluster confcheck filepath`	Check the cluster configuration, with the content of the xml file passed as argument, without applying it

`safekit cluster confinfo`	Return, for each active cluster node:
	• the date of last cluster configuration,
	• the digital signature of last cluster configuration
	• the state: locked (1) or unlocked (0) status for the cluster configuration
	
	This command allows checking if all node of a cluster have the same configuration.
	Ex:
	`safekit cluster conf info`
	`Node        Signature              Date                Lock`
	`rh6server7  6f1032b11a7b2 … 33e67c 2016-05-20T17:06:45  0`
	`rh7server7  6f1032b11a4e0 … 33e67c 2016-05-20T17:06:45  0`
	
	Note The SafeKit cluster configuration must be the same on all nodes of a cluster. Asymmetric cluster configurations are not supported.

`safekit cluster deconfig`	Remove the cluster configuration and the cryptographic key.
`safekit cluster state`	Return the global SafeKit modules configuration state For each installed module on each cluster node, this commands list: • the node name, • module name, • module mode (farm or mirror) • internal module id number, • date of last module configuration, • digital signature of last configuration This command list which modules are installed on which nodes of the cluster. Signature and date of last configuration on each node allow checking that a module has the same configuration on all nodes, and if not, which node has the most recent configuration.
`safekit cluster genkey`	Create cryptographic key for global SafeKit communication (implemented in the `safeadmin` process). The cluster configuration must be deployed again (with `safekit -G`) for this command to take effect.
`safekit cluster delkey`	Suppress cryptographic keys for global SafeKit communication. The cluster configuration must be applied again (with `safekit -G`) for this command to take effect.

`safekit -H “[http],*” -G`	Redo a name resolution for all names specified in cluster.xml and `userconfig.xml` of modules, without stopping modules (when possible).
`safekit -H <url>[,<url>] -G`	Distributes the local cluster configuration and associated cryptographic key if it exists, to the target nodes specified in the URL list. Ex: `safekit -H http://192.168.1.1:9010,http://192.168.1.2:9010 -G`

9.4 Command lines to control modules

The commands apply to the module named `AM`, passed as an argument with the `-m` option.

`safekit start -m AM`	Starts the module
`safekit waitstart -m AM`	Waits for the end of the module start
`safekit stop -m AM`	Stops the module
`safekit waitstop -m AM`	Waits for the end of the module stop
`safekit waitstate -m AM STOP | ALONE | UP | PRIM | SECOND`	Wait for the required stable state (red or green).
`safekit restart -m AM`	Executes only application stop and start scripts Note For mirror modules, there is no failover on the other server if the module is `PRIM`
`safekit swap [nosync] -m AM`	Mirror modules only Swaps the roles of primary and secondary nodes. Use `nosync` to swap without synchronizing the replicated directories.
`safekit stopstart -m AM`	Unlike the `safekit restart -m AM` command, the `safekit stopstart -m AM` command causes a complete stop of the module followed by a start. If the module was `PRIM`, there is a failover of the `PRIM` module on the other server Note Equivalent to `safekit stop -m AM; safekit start -m AM`

`safekit prim -m AM`	Mirror modules only Forces the module to start as primary. It fails if the other server is already primary. The main use case of this command is described in section 5.3 page 97
`safekit second [fullsync] -m AM`	Mirror modules only Forces the module to start as secondary. It fails if the other server is not primary. Use `fullsync` to force the full synchronization of the replicated directories.
`safekit forcestop -m AM`	Forces the module stop even if some resources are frozen

`safekit errd suspend -m AM`
`safekit errd resume -m AM`
	Suspends/resumes the error detection of module processes defined in <errd> section of `userconfig.xml`
	Useful if you want to stop the application without changing the module state.
	The resource variable *usersetting.errd* reflects the current setting.

`safekit checker off -m AM`
`safekit checker on -m AM`
	Used to stop or start all checkers (interface, TCP, IP, custom, etc ...)
	Useful for maintenance operation, when man knows that some checker will detect a problem because some parts of the IT infrastructure will be stopped, and don't want that Safekit start a failover.
	Notes:
	✓ could be used only on a live module in a stable state (`ALONE`, `UP`, `PRIM`, `SECOND`, `WAIT`)
	✓ the resource variable *usersetting.checker* reflects the current setting
	✓ a side effect of this command is the execution of the update command.

`safekit failover off -m AM`
`safekit failover on -m AM`
	Used to dynamically set the failover attribute to on or off (see section 13.2.3 page 237).
	Notes:
	✓ could be used only on a mirror live module in a stable state (`ALONE`, `PRIM`, `SECOND`,`WAIT`).
	✓ this command must be issued on all machines belonging to the same cluster to not have unexpected results.
	✓ the resource variable *usersetting.failover* reflects the current setting.
	✓ a side effect of this command is the execution of the update command.

9.5 Command lines to monitor modules

The commands apply to the module named `AM`, passed as an argument with the `-m` option.

`safekit level [-m AM]`	Indicates the version of SafeKit and the license With the `AM` parameter, the "level" script of the module is called, and its results displayed
`safekit state`	Displays the status of all modules

`safekit state -m AM`
`[-v | -lq]`
	Displays the status of the AM module
	With the verbose option `-v`, status of all the module resources are listed: see the usefulness of resources in section 7.9 page 117
	With the option `-lq`, the command returns status (and exit code): `STOP (0), WAIT (1), ALONE (2), UP (2), PRIM (3), SECOND (4)`

`safekit log -m AM [-s nb]`
`[ -A | -I] [-l en|fr]`
	Displays the last `nb` E(vent) messages of the AM module log.
	Use `-I` option for displaying also I(nformation) messages, or `-A` for displaying all messages (including debug ones).
	Use `-l` option for choosing the language, `en`(glish) or `fr`(ench).
	Default: `-s 300`

`safekit logview -m AM [-A | -I] [-l en|fr]`	View in real time the last E(vent) messages of the AM module log. Use `-I` option for displaying also I(nformation) messages, or `-A` for displaying all messages (including debug ones). Use `-l` option for choosing the language, `en`(glish) or `fr`(ench).
`safekit logview -m AM -s 300 [-A | -I] [-l en|fr]`	View in real time the AM module log messages starting from the last 300 messages
`safekit logsave -m AM [-l en|fr] [-A] /tmp/f.txt`	Save E(vent) messages of the AM module log in /tmp/f.txt (absolute path mandatory). Use `-I` option for saving also I(nformation) messages, or `-A` for saving all messages (including debug ones). Use `-l` option for choosing the language, `en`(glish) or `fr`(ench).
`safekit printi|printe -m AM "message"`	Application start/stop scripts can write messages in the module log with I or E level.

9.6 Command lines to configure modules

`safekit config -m AM`	Apply changes made in `SAFE/modules/AM`: `userconfig.xml`, `start_prim/both` or `stop_prim/both` (mirror/farm) Makes each plug-in defined in `userconfig.xml` <errd>, <vip>, <rfs>, <user>... considered in the new module configuration This command could be run on a server in the stable states `STOP`, `ALONE` or `WAIT` (red). In `STOP` state all the configuration parameters could be modified. Some configuration parameters can be changed while the module is running in `ALONE` or `WAIT` (red) states. This feature is called *dynamic configuration*. Parameters that could be dynamically changed are reported into section 13 page 235 that describes all configuration parameters.
`safekit module genkey -m AM`	Generates cryptographic keys for the module instances network exchanges encryption. Considered after the next configuration of the module.
`safekit module delkey -m AM`	Erase cryptographic keys associated with the module. After the next configuration, module instances network exchanges will be performed without encryption.

`safekit -H <url>[,<url>] -E AM`	Distributes the local configuration for the module `AM` and associated cryptographic key if it exists, to the target nodes specified in the URL list. Ex: `safekit -H http://192.168.1.1:9010,http://192.168.1.2:9010 -E mirror`
`safekit confinfo -m AM`	Display information on the active and current configuration of the module `AM`. ⇨ the active configuration is the last configuration successfully applied. It is in `SAFE/private/modules/AM` ⇨ the current configuration is the one located in `SAFE/modules/AM`. It may be different from the active one when it has been modified and not yet been applied This command is useful for checking the configuration of the module. It displays: ⇨ the signature value and a last modification date (Unix timestamp) for the active configuration ⇨ the signature value and last modification date (Unix timestamp) for the current configuration When the signature values are different, it means that the configurations are not identical and that you may have to apply the current configuration. You can run this command on all the cluster nodes that implement the module to check that the configuration of the module is identical on all nodes.
`safekit confcheck -m AM`	Check the module configuration under `SAFE/modules/AM` without applying
`safekit module install -m AM [-M id] [-r] SAFE/Application_Modules/AM.safe`	Installs the AM.safe module under the `AM` name `[-r]` force reinstallation of the module `[-M id]` forces the installation of the module with the `id` specified as module id
`safekit module package -m AM /…/newAM.safe`	Packages the `AM` module in `/…/newAM.safe` (absolute path mandatory) Used by the console to create a backup in `SAFE/Application_Modules/backup/`
`safekit module uninstall -m AM`	Uninstalls the `AM` module. Deletes the module configuration directory `SAFE/modules/AM`
`safekit module list`	Lists the names of the installed modules

`safekit module listid`	Lists the names and ids of the installed modules
`safekit module getports -m AM (or -i id)`	Lists the communication ports used by the module to communicate between servers

9.7 Command lines for support

`safekit snapshot -m AM /tmp/snapshot_xx.zip`	Saves the snapshot of the `AM` module in /tmp/snapshot_xx.zip (absolute path mandatory) A snapshot creates a dump and gathers under `SAFEVAR/snapshot/modules/AM` the last 3 dumps and last 3 configurations to collect them in a .zip file To analyse snapshots, see 7.16 page 123 To send snapshots to Evidian support, see 8 page 133
`safekit dump -m AM`	To solve a problem in real time on a server, make a dump of the `AM` module A dump creates a directory `dump` `dump_year_month_day_hour_mn_sec` on the server side under `SAFEVAR/snapshot/modules/AM`. The `dump` directory contains the module log and status, as well as information on the system state and SafeKit processes at the time of the dump
`safekit -r "specialcommand"`	Calls the special command in `SAFEBIN` with SafeKit environment variables set.

`safekit clean [all | log | process | resource] [-m AM]`	Clean the logs, the resource file, and the main processes of the module `AM`.
	Important: This command must be used with caution since it deletes working files and kills processes.
	`safekit clean log -m AM`
	Clean the logs (verbose and not verbose logs) of the module. To be used when these logs are corrupted (e.g.: errors in log view).
	`safekit clean resource -m AM`
	Reinitialize the resource file of the module. To be used when this file is corrupted (e.g.: errors in resources display)
	`safekit clean process -m AM`
	Kill the main processes (`heart`) of the module. To be used when the `stop` and `forcestop` of the module did not achieve to kill these processes.
	`safekit clean all -m AM`
	Default value. Clean log, resource, and process.

10.Advanced administration

10.1 SafeKit environment variables and directories

10.1.1 Global

Variable	**Description**
`SAFE` (given by `safekit -p`)	SafeKit installation directory: **`SAFE=/opt/safekit`** on Linux and **`SAFE=C:\safekit`** on Windows if SystemDrive=C: The license is under **`SAFE/conf/license.txt`**
`SAFEVAR` (given by `safekit -p`)	SafeKit working files directory: **`SAFEVAR=C:\safekit\var`** on Windows and **`SAFEVAR=/var/safekit`** on Linux
`SAFEBIN` (given by `safekit -p`)	SafeKit binary installation directory: **`C:\safekit\private\bin`** on Windows and **`/opt/safekit/private/bin`** on Linux. Useful to access SafeKit special commands (see 14.4 page 296)
`SAFE/Application_Modules`	Installable .safe modules directory. Once a module has been installed, the module is located under **`SAFE/modules`**

10.1.2 Module

Variable	Description
`SAFEMODULE`	The name of the module. The `safekit` command no longer needs the module name parameter (`-m AM = -m SAFEMODULE`)
`SAFE/modules/AM and SAFEUSERBIN`	Editing a module, named AM, and its scripts is made inside directory **`SAFE/modules/AM`**. There are **`userconfig.xml`** file and application start and stop scripts **`start_prim`**, **`stop_prim`** for a mirror, **`start_both`**, **`stop_both`** for a farm (online edition or through the SafeKit console) After a module configuration (`safekit config -m AM` or web console/ Advanced Configuration /Installed modules/ module/ Apply the configuration or web console/ Configuration/ ▼ on the module/ Edit the configuration), scripts are copied to the runtime directory **`SAFE/private/modules/AM/bin`**: this is the value of `SAFEUSERBIN` (do not modify scripts at this place)
`SAFEVAR/modules/AM and SAFEUSERVAR`	Module, named A, working files directory (`SAFEUSERVAR=`**`SAFEVAR/modules/AM`**) Output messages of application scripts are in **`SAFEVAR/modules/AM/userlog.ulog`.** To check if there are errors during start or stop of the application. Be careful, sometimes the userlog is disabled because it is too large with <user logging="none"> in `userconfig.xml` of the module Note Since SafeKit 7.4.0.19, the extension for the application log file has changed. The file name is now `userlog.ulog` and it was `userlog.AM`.
`SAFEVAR/snapshot/modules/AM`	Directory of dumps and configurations put in a snapshot of the module named AM. See section 9.7 page 151 that describes command lines for support.

The module tree (packaged into a .safe or installed into `SAFE/modules/AM`) is the following:

`AM`	Application module name
`conf`	
`userconfig.xml`	User XML configuration file
`userconfig.xml.template`	Internal use only
`modulekey.p12`	Optional. Internal use only (encryption of the module internal communications)
`modulekey.dat`	Optional. Internal use only (encryption of the module internal communications)
`bin`	
`prestart`	User script executed on module start
`start_prim or start_both`	User script to start the application in mirror or farm module
`stop_prim or stop_both`	User script to stop the application in mirror or farm module
`poststop`	User script executed on module stop
`web`	
`index.html`	File for the SafeKit web console
`manifest.xml`	Internal use only

`index.html` is an HTML page with JavaScript that is displayed in the web console/Configuration wizard (described in 3.3.2.2 page 47). You can modify this page to customize the Edit the Configuration form in the configuration wizard. When `index.html` is not present (in previous versions modules for instance), the web console instead proposes to edit the `userconfig.xml` file.

10.2 SafeKit processes and services

SafeKit Services	**Processes per module**	
`safeadmin` (`safeadmin` process): main and mandatory service	`heart`: manages the recovery procedures	`vipd`: synchronizes a farm of servers

SafeKit Services	Processes per module	
safewebserver (httpd process): service for the console, for <module> checkers and the distributed commands	errd: manages detection of process death	nfsbox, nfsadmin, reintegre: file replication and reintegration
safeagent (safeagent process): SafeKit SNMP agent (optional)	checkers (ipcheck, intfcheck, …)	

See 10.3.3.1 page 158 and 10.3.3.2 page 159 for full details on SafeKit processes name and ports used.

10.3 Firewall settings

If a firewall is active on the SafeKit server, you must add rules to allow network traffic:

- between servers for internal communication (global runtime and module specific)
- between servers and workstations running the SafeKit console

10.3.1 Firewall settings in Linux

If you opted-in for automatic local firewall configuration during SafeKit installation, you do not have to apply the following procedures, except for configuring the safeagent service for SafeKit SNMP agent is enabled.

If you opted-out for automatic local firewall configuration, you must configure the firewall manually or you may use the firewallcfg command (in SAFEBIN). It inserts (or remove) the firewall rules required by the SafeKit core processes (safeadmin and safewebserver services) and modules processes to communicate with their peers in the cluster.

Administrators should review the script for conflicts with local policy before applying it.

firewallcfg add firewallcfg del	Add (or delete) the firewalld or iptable firewall rules for the SafeKit safeadmin and safewebserver services SAFEBIN=/opt/safekit/private/bin ⇨ SAFEBIN/firewallcfg add add firewall rules for safeadmin and safewebserver ⇨ SAFEBIN/firewallcfg del delete firewall rules for safeadmin and safewebserver

`firewallcfg add AM`
`firewallcfg del AM`
	Add (or delete) the firewalld or iptable firewall rules for the SafeKit modules
	`SAFEBIN=/opt/safekit/private/bin`
	➩ `SAFEBIN/firewallcfg add AM`
	add firewall rules for the module named `AM`
	Important This command must be applied after the first configuration of the module, and on next configurations if used ports have changed (check it with the command `safekit module getports -m AM`).
	➩ `SAFEBIN/firewallcfg del AM`
	delete firewall rules for the module named `AM`

`firewallcfg add safeagent`
`firewallcfg del safeagent`
	Add (or delete) the firewalld or iptable firewall rules for the SafeKit `safeagent` service
	`SAFEBIN=/opt/safekit/private/bin`
	➩ `SAFEBIN/firewallcfg add safeagent`
	add firewall rules for `safeagent`
	Important This command must be applied when you enable the SafeKit SNMP agent.
	➩ `SAFEBIN/firewallcfg del safeagent`
	delete firewall rules for `safeagent`

10.3.2 Firewall settings in Windows

When using the operating system firewall (Microsoft firewall), you may use the `firewallcfg` command (in `SAFEBIN`). It inserts (or remove) the firewall rules required by the processes of SafeKit services (`safeadmin`, `safewebserver`, `safeagent`, `safeacaserv`) and modules processes to communicate with their peers in the cluster.

Administrators should review the script for conflicts with local policy before applying it.

`firewallcfg add`
`firewallcfg del`
	Add (or delete) the Microsoft firewall rules
	`SAFEBIN=C:\safekit\private\bin` (if `%SYSTEMDRIVE%`=C:)
	cd SAFEBIN
	`firewallcfg add`
	add firewall rules for SafeKit core and modules processes
	➩ `cd SAFEBIN`
	`firewallcfg del`
	delete firewall rules for SafeKit core and modules processes

10.3.3 Other firewalls

If you use another firewall or want to check rules against local policy, the following lists processes and ports used by SafeKit services and modules that may be useful to configure the firewall.

10.3.3.1 List of processes

10.3.3.1.1 Processes performing local-only network exchanges

- Processes for a mirror module
  - ✓ `errd`: manages detection of process death
  - ✓ `nfsadmin`, `nfscheck`: manage the file replication

- Processes for a farm module
  - ✓ `errd`: manages detection of process death
  - ✓ `heart`: manages the recovery procedures

10.3.3.1.2 Processes performing external network exchanges

- Processes common to all the SafeKit servers, one process by server, started at boot:
  - ✓ `safeadmin` service (`safeadmin` process)

    main and mandatory administration service
  - ✓ `safewebserver` service (`httpd` process)

    web service for the console, for <module> checkers and the distributed commands
  - ✓ `safecaserv` (`httpd` process)

    web service for securing the web console with the SafeKit PKI (optional)
  - ✓ `safeagent` service (`safeagent` process)

    SafeKit SNMP v2 agent (optional)

- Processes for a mirror module (depending on its configuration):
  - ✓ `heart`: manages the recovery procedures
  - ✓ `arpreroute`: manages arp requests (sends ARP packet)
  - ✓ `nfsbox`, `reintegre`: manage the file replication and reintegration
  - ✓ `splitbraincheck`: manage the splitbrain detection (sends ICMP ping packets)

- Processes for a farm module (depending on its configuration):
  - ✓ `vipd`: synchronizes a farm of servers
  - ✓ `arpreroute`: manages arp requests (sends ARP packet)

- Processes for a mirror or a farm module depending on checkers configuration:
  - `intfcheck`: for checking interface (interface checker configuration automatically generated when <interface check=on>)
  - `pingcheck`: for pinging an address (<ping> configuration)
  - `ipcheck`: for checking a locally defined ip address (virtual ip checker automatically generated when `<virtual_addr check=on>`)
  - `modulecheck`: for checking a SafeKit module (`<module>` configuration)
  - `tcpcheck`: for checking a TCP connection (`<tcp>` configuration)

10.3.3.2 List of ports

The following list ports used by SafeKit services and modules.

10.3.3.2.1 Ports used by services

- `safeadmin`

  By default, remote access on UDP port 4800 (to communicate with `safeadmin` instances on other SafeKit servers) and local access on UDP port 6259.

  For changing the remote port value, see section 12.1.3 page 229.

  The local port value is defined by the attribute `mapper` into the global SafeKit configuration file `safeini.xml` (in Linux: `/etc/safeini.xml`; in Windows: `c:\Windows\safeini.xml`).

  Note

  Before upgrading SafeKit, save this file if you have modified it because its content is not preserved.

- `safewebserver`

  Local and remote TCP access, by default, on port 9010 for HTTP or port 9453 for HTTPS. For the ports value definition, see section 10.6 page 167.

  This service is accessed locally and from remote SafeKit servers and remote workstation running the SafeKit console.

- `safecaserv` (optional)

  Local and remote access on TCP port 9001 by default. For the port value definition, see section 11.6.5 page 225.

  This service is accessed locally, and from remote SafeKit servers and remote workstation running the HTTPS configuration wizard with the SafeKit PKI.

- `safeagent` (optional)

  Local and remote access on UDP port 3600 by default. For the port value definition, see section 10.8 page 173.

10.3.3.2.2 Ports used by modules

When a module is configured on a SafeKit server, you can run the command `safekit module getports -m AM` to list the external ports used by the module `AM`. For firewall configuration, you must configure all SafeKit servers to enable communications targeted at these ports.

The ports values for one module are automatically computed depending on its module id. Run the command `safekit module listid` to list all the installed modules with their name and id.

You can run the command `safekit module getports -i ID` to list the ports that could be used by a module that got the id value ID (this command can be run even if the module is not yet installed, but it will return a superset of the really used port by the module).

The following gives rules for computing ports values depending on the module id. When checkers are configured for the module, you may also need to change the firewall configuration according to the checkers configuration. You must enable all communications on localhost between SafeKit processes.

➾ For a mirror module:

✓ Port used by heart
UDP port used for sending heartbeats between SafeKit servers
port=8888 +(id-1)

✓ Ports used by rfs (file replication)
TCP port used for replications requests between SafeKit servers
safenfs_port=5600 +(id-1)x4

Example for a mirror module with id 1

```
safekit module getports -m mirror


List of the ports used by SafeKit

Process         Ports
safeadmin
        port    UDP 4800

webconsole
        port    TCP 9010
heart
        port    UDP 8888
rfs
        safenfs_port    TCP 5600
```

➾ For a farm module
✓ Port used by farm
UDP port used for communications between all SafeKit nodes
port   4803 + (id-1)x3

Example for a farm module with id 2

```
safekit module getports -m farm

List of the ports used by SafeKit

Process         Ports
safeadmin
        port    UDP 4800
webconsole
        port    TCP 9010
farm
        port    UDP 4806
```

➾ For configured checkers

- ✓ Ping checker for mirror or farm module
  Change ICMP settings to allow ping at destination to the address defined into the configuration.
- ✓ TCP checker for mirror or farm module
  Allow TCP connections at destination to the address defined into the <tcp> configuration if this address is not local.
- ✓ Module checker
  Allow TCP connections at destination to 9010 port of the node running the module that is checked.
- ✓ Splitbrain checker
  Change ICMP settings to allow ping at destination to the witness defined into the <splitbrain> configuration.

10.4 Boot and shutdown setup in Windows

`safeadmin` service is configured for automatically starting on boot and stopping on shutdown. In turn, this service starts modules configured for starting at boot and shutdown all modules.

On some Windows platforms, the `safeadmin` boot start fails because the network configuration is not ready, and the modules shutdown does not have time to complete since the timeout for services shutdown is too short. If you encounter such problems, apply one of the following procedures.

Important

When using the SNMP agent, adapt the following procedures to set the manual start of the `safeagent` service and include its start/stop into SafeKit start-up (`safekitbootstart.cmd`) and shutdown (`safekitshutdown.cmd`) scripts.

10.4.1 Automatic procedure

You can run the script as follow:

1. open a PowerShell window as administrator
2. `cd SAFE\private\bin`

3. run `addStartupShutdown.cmd`

This script sets the manual start for `safeadmin` service and adds default SafeKit start-up (`safekitbootstart.cmd`) and shutdown (`safekitshutdown.cmd`) scripts as part of the computer group policy start-up/shutdown scripts. If the script fails, apply the manual procedure below.

10.4.2 Manual procedure

You must apply the following procedure that uses the Group Policy Object Editor.

1. set manual start for `safeadmin` service
2. start the MMC console with the `mmc` command line
3. File - Add/Remove Snap-in Add - "Group Policy Object Editor" – OK
4. under "Console Root"/"Local Computer Policy"/"Computer Configuration"/"Windows Settings"/"Scripts (Start-up/Shutdown)", double click on "Start-up". Click on Add then set for "Script Name:" `c:\safekit\private\bin\safekitbootstart.cmd`. This script launches the `safeadmin` service.
5. under "Console Root"/"Local Computer Policy"/"Computer Configuration"/"Windows Settings"/"Scripts (Start-up/Shutdown)", double click on "Shutdown". Click on Add then set for "Script Name:" `c:\safekit\private\bin\safekitshutdown.cmd`. This script shutdowns all running modules.

10.5 Securing module internal communications

You can secure communications for the module between cluster nodes by creating cryptographic keys associated with the module. By default, these keys are generated by SafeKit with a “private” certification authority (SafeKit PKI). In SafeKit <= 7.4.0.31, the generated key has a validity period of 1 year. See section 10.5.3.1 page 165 for solutions when the key expires.

Since SafeKit 7.4.0.16, you can also provide your own certificates generated with your trusted certification authority (enterprise PKI or commercial PKI). See section 10.5.3.2 page 166 for details.

Since SafeKit 7.4.0.32, the module can be reconfigured with new keys while it is in ALONE state (dynamic update).

Important

When encryption is not properly configured (e.g.: not the same key on all cluster nodes of the module), the module internal communications between nodes are rejected. In this case, the module configuration is not identical on all nodes. You must apply again the configuration on all nodes.

You can check the configuration by running on each node the command `safekit confinfo –m AM` where AM is the module name (see section 9.6 page 149). This information is also displayed by the SafeKit web console before editing the configuration of the module and before running a global start.

Important

When encryption is not properly configured (e.g.: not the same key on all cluster nodes of the module), the module internal communications between nodes are rejected. In this case, the module configuration is not identical on all nodes. You must apply again the configuration on all nodes.

You can check the configuration by running on each node the command `safekit confinfo -m AM` where AM is the module name (see section 9.6 page 149). This information is also displayed by the SafeKit web console before editing the configuration of the module and before running a global start.

Note

The **`encryption`** resource reflects the current communication mode of the module: “on”/”off” when encryption is active/not active. To see the resources state, display web console/✓ Control/Select the node/Resources tab.

Since SafeKit 7.5, the resource name is `usersetting.encryption`.

10.5.1 Configuration with the SafeKit Web console

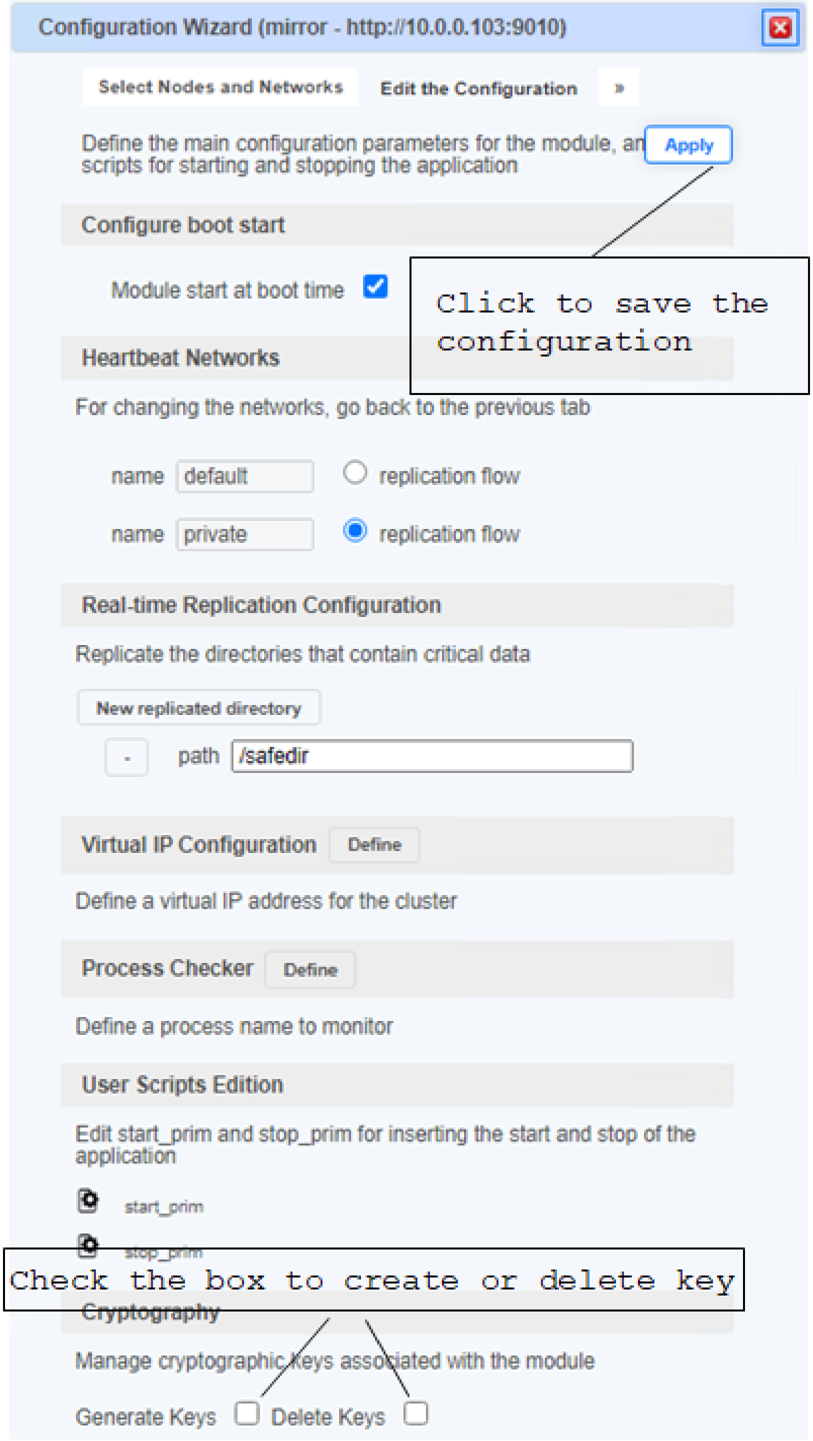
When configuring the module with the SafeKit web console (see section 3.3 page 41):

- ➾ In Configuration wizard
- ➾ Edit the configuration tab
- ➾ Fill in the form
- ➾ (1) Check the box Generate Keys for creating cryptographic key

or

Check the box Delete Keys for removing cryptographic key

- ➾ (2) click on the Apply button to save changes and go to next step for applying it on all nodes of the module



Configuration Wizard (mirror - http://10.0.0.103:9010)

Define the main configuration parameters for the module, an scripts for starting and stopping the application

Click to save the configuration

Configure boot start

Module start at boot time ☑

Heartbeat Networks

For changing the networks, go back to the previous tab

name default ○ replication flow

name private ● replication flow

Real-time Replication Configuration

Replicate the directories that contain critical data

path /safedir

Virtual IP Configuration

Define a virtual IP address for the cluster

Process Checker

Define a process name to monitor

User Scripts Edition

Edit start_prim and stop_prim for inserting the start and stop of the application

Check the box to create or delete key

Cryptography

Manage cryptographic keys associated with the module

Generate Keys □ Delete Keys □

10.5.2 Configuration with the Command Line Interface

The commands line equivalent for configuring a module, named AM, with cryptographic key are:

1. Stop the AM module on all nodes
2. On one node, log as administrator/root and open a command shell window
3. Run `safekit module genkey -m AM`
4. Run `safekit -H "server1,server2" -E AM`

   where server1 and server2 are the nodes that implement the module

The commands line equivalent for re-configuring a module without cryptographic key are:

1. Stop the AM module on all nodes
2. On one node, log as administrator/root and open a command shell window
3. Run `safekit module delkey -m AM`
4. Run `safekit -H "server1,server2" -E AM`

   where server1 and server2 are the nodes that implement the module

For more details on commands, refer to section 9.6 page 149.

10.5.3 Advanced configuration

10.5.3.1 Advanced configuration with the SafeKit PKI

In SafeKit <= 7.4.0.31, the key for encrypting the module communication has a validity period of 1 year. When it expires in a mirror module with file replication, the secondary fails to reintegrate. You must re-configure the module with a new key, as explained in SK-0084, for reverting to normal behavior. In SafeKit > 7.4.0.31, the validity period has been set to 20 years.

If you cannot upgrade SafeKit, you can generate new keys with a longer validity period. For this apply the following procedure:

1. Stop the AM module on all nodes
2. On one node, log as administrator/root and open a command shell window
3. Run `safekit module genkey -m AM`
4. Delete the file `SAFE/modules/AM/conf/modulekey.p12`
5. Change to the directory `SAFE/web/bin`
6. Run

   ```
   ./openssl req -config ../conf/ssl.conf -subj
   "/O=SafeKiModule/CN=mirror" -new -x509 -sha256 -nodes -days 3650 -newkey
   rsa:2048 -keyout pkey.key -out cert.crt
   ```

   Set the `-days` value to the validity period you want

7. Run `./openssl pkcs12 -export -inkey ./pkey.key -in ./cert.crt -name "Module certificate" -out modulekey.p12`

   This command requires to fill a password. Contact Evidian support to get the correct value for the password

8. Delete the files `pkey.key` and `cert.crt`

9. Move the file `modulekey.p12` into `SAFE/modules/AM/conf`

10. Run `safekit -H "server1,server2" -E AM`

    where server1 and server2 are the nodes that implement the module

The module is configured, on the 2 nodes, with the new key and ready to start.

10.5.3.2 Advanced configuration with your PKI

Since SafeKit 7.4.0.16, you can provide your own key generated with your trusted certification authority (enterprise PKI or commercial PKI). For this apply the following procedure:

1. Stop the AM module on all nodes

2. On one node, log as administrator/root and open a command shell window

3. Run `safekit module genkey -m AM`

4. Delete the file `SAFE/modules/AM/conf/modulekey.p12`

5. Append the X509 certificate in PEM format, for your certification authority (certificate of the CA or certificate bundle of all the certificate authorities) to the file `SAFE/web/conf/cacert.crt`

6. Change to the directory `SAFE/web/bin`

7. Generate your certificate with your PKI with the subject set to `"/O=SafeKiModule/CN=mirror"`

8. Copy the generated files `pkey.key` and `cert.crt` into the directory `SAFE/web/bin`

9. Run `./openssl pkcs12 -export -inkey ./pkey.key -in ./cert.crt -name "Module certificate" -out modulekey.p12`

   This command requires to fill a password. Contact Evidian support to get the correct value for the password

10. Delete the files `pkey.key` and `cert.crt`

11. Move the file `modulekey.p12` into `SAFE/modules/AM/conf`

12. Run `safekit -H "server1,server2" -E AM`

    where server1 and server2 are the nodes that implement the module

The module is configured, on the 2 nodes, with the new key and ready to start.

10.6 Configuration of the SafeKit web service

SafeKit comes with a web service, `safewebserver`, which runs on each SafeKit server. It is a standard Apache web service that is **mandatory** for running:

- ➾ the web console (see section 3 page 35)
- ➾ the distributed command line interface (see 9.1 page 141)
- ➾ the <module> checkers (see 13.16 page 285)

`safewebserver` starts automatically at the end of SafeKit package install and on server reboot. If you do not need the SafeKit web service and want to remove the automatic boot start, refer to section 9.2 page 142.

Since SafeKit 7.5, the default configuration is HTTP with file-based authentication, initialized with a single `admin` user that got the Admin role. If you want to change it, refer to section 11 page 177.

10.6.1 Configuration files

The configuration of an instance of `safewebserver` on a SafeKit server is contained in the `SAFE/web/conf` directory. It consists in standard Apache configuration files (see http://httpd.apache.org). The configuration is split into many files that are included or not depending on desired settings.

Important

After changes, you have to restart the service with the command: `safekit webserver restart` (see section 9.2 page 142).

If necessary, you should only modify the main configuration file `httpd.conf` to suit your needs. Comment character # disables the definition. The file contains the definition of:

Connection port definition:

- ➾ HTTP port

```
httpadminport (9010)
#httpcontrolport (9010)
#httpmonitorport (9010)
```

Admin role by default. Depending on the desired role, uncomment the corresponding port and comment the others:

- ✓ `httpadminport` for Admin role
- ✓ `httpcontrolport` for Control role
- ✓ `httpmonitorport` for Monitor role

When uncommenting `httpcontrolport` or `httpmonitorport`, user authentication must be disabled (see below).

- ➾ HTTPS port

```
httpsport (9453)
```

User authentication definition:

➩ File-based authentication

```
Define usefile
```

Enabled by default. Comment to disable.

- ✓ when using `httpcontrolport` or `httpmonitorport`, it must be disabled
- ✓ when enabled, `httpadminport` must be enabled and `useldap` must be disabled

➩ LDAP/AD authentication

```
# Define useldap
…
```

Disabled by default. Uncomment to enable.

- ✓ when using `httpcontrolport` or `httpmonitorport`, it must be disabled
- ✓ when enabled, `httpadminport` must be enabled and `usefile` must be disabled

Apache logging definition:

It is disabled by default.

```
#Define Loglevel info
#Define accesslog
```

Uncomment these lines to enable the logging for debug purposes. Logging files `httpd.log` and `access.log` are in `SAFEVAR`.

The other configuration files are listed below. Modifying one of them may cause problems when upgrading SafeKit.

Global configuration	**`httpd_main.conf`**
HTTP configuration	**`httpd.webconsole.conf`**
HTTPS configuration and client certificate user authentication	**`httpd.webconsolefileauth.conf`** ➩ using `sslgroup.conf` files into `SAFE/web/conf`
Form authentication configuration	**`httpd.webconsoleformauth.conf`** ➩ for HTTP or HTTPS
File-based authentication configuration	**`httpd.webconsolefileauth.conf`** ➩ for HTTP or HTTPS

	⇨ using `user.conf` and `group.conf` files into `SAFE/web/conf`
LDAP/AD authentication configuration	**`httpd.webconsoleldap.conf`** ⇨ for HTTP or HTTPS ⇨ using a LDAP/AD server

The HTTP and HTTPS configurations cannot be active simultaneously.

Do not edit `.default` files in `SAFE/web/conf` since they are backups of delivered configuration files.

10.6.2 Connection ports configuration

By default, connect the web console with the URL `http://servername:9010`. The SafeKit web server will redirect to the appropriate page according to your security settings.

If you need to change the default value:

1. Edit `SAFE/web/conf/httpd.conf` and change the value of `httpadminport`, `httpcontrolport`, `httpmonitorport` or `httpsport` variables.
2. Restart the service using the command `safekit webserver restart.`

The default value `9010(HTTP)/9453(HTTPS)` is also used by other SafeKit components. Therefore, if the default value is changed, the configuration for these components must also be changed, as follows:

- in the global SafeKit configuration file `safeini.xml`, for the distributed commands:
    1. Edit the file `safeini.xml` (in Linux: `/etc/safeini.xml`; in Windows: `c:\Windows\safeini.xml`)
    2. Remove the strings <-- and --> that comment the `SAFESRVPORT` definition
    3. Replace the value of `SAFESRVPORT` by the new value that you have defined in `httpd.conf`

Note

Before upgrading SafeKit, save this file if you have modified it because its content is not preserved.

- in the configuration of modules that define a <module> checker:
    1. Edit the module configuration file `userconfig.xml`
    2. Add the `port` attribute and assign it to the new port value

```
<check>
    <module name="mirror">
     <to addr="192.168.1.31" port="9010"/>
    </module>
  </check>
```

3. Apply the new configuration of the module

10.6.3 HTTP configuration

The default configuration is for HTTP.

The default configuration is also set with file-based authentication, initialized with a single `admin` user that got the Admin role. This one can be extended for other users or roles ; or replaced by another configuration. For a detailed description, see section 11 page 177.

10.6.4 HTTPS configuration

The HTTPS configuration requires the installation of certificates and the definition of user authentication as described in section 11 page 177. Once done, HTTPS configuration can be enabled:

1. copy `SAFE/web/conf/httpd.webconsolessl.conf` into the `SAFE/web/conf/ssl` directory
2. restart the service using the command `safekit webserver restart`

Skip this procedure if you use the HTTPS configuration wizard since it applies it automatically.

10.6.5 HTTPS <-> HTTP configuration

To re-enable the HTTP configuration if it has been changed to HTTPS:

1. remove the file `SAFE/web/conf/ssl/httpd.webconsolessl.conf`
2. restart the service using the command `safekit webserver restart`.

All the files necessary for the HTTPS configuration are preserved. It is therefore possible to revert to the HTTPS configuration if necessary:

1. copy `SAFE/web/conf/httpd.webconsolessl.conf` into the `SAFE/web/conf/ssl` directory
2. restart the service using the command `safekit webserver restart`

10.7 Mail notification

For mail notification, you have first to choose a command line program to send mail. For Windows, you can download windows binary from the mailsend download area. For Linux, you can use the `mail` command instead of `mailsend`.

Important

Mail notification is implemented thanks to user scripts of the module. These scripts can be edited with the SafeKit console or on the server side. Each time you modify one of these scripts, you must re-apply the module configuration on all nodes (via the SafeKit console or the command).

10.7.1 Mail notification on the start and the stop of the module

The following lines, inserted into at the end of the `prestart` script of a module (named `AM`), send an e-mail with the name of the module and server on which the module is started:

⇨ In Windows: **`c:\safekit\modules\AM\bin\prestart.cmd`**

```
if [%3] NEQ [start] goto nostart
rem send mail only on start (not on stopstart or stopwait)
FOR /F "usebackq" %%i IN (`hostname`) DO SET HOSTNAME=%%i
mailsend.exe -d mydomain.com -smtp smtp.mydomain.com
-t admin@mydomain.com -f SafeKit -sub "Start module %SAFEMODULE% on %HOSTNAME%" -
M "Running prestart" +cc +bc
:nostart
```

⇨ In Linux: **`/opt/safekit/modules/AM/bin/prestart`**

```
if [ "$3" = "start" ]; then
  # send mail only on start (not on stopstart or stopwait)
  echo "Running prestart" | mail -s " Start module $SAFEMODULE on `hostname`"
admin@mydomain.com
fi
```

When the module is stopping, it can be notified using the `poststop` script. This one is not delivered by default and can be created as follow (for the module named `AM`):

⇨ In Windows: **`c:\safekit\modules\AM\bin\poststop.cmd`**

```
@echo off
rem Script called on module stop, stopstart, stopwait
rem For logging into module log use:
rem "%SAFE%\safekit" printi | printe "message"
rem stdout goes into Application log
echo "Running poststop %*"

rem send an email only on stop (not on stopstart or stopwait)
if [%3] NEQ [stop] goto nostop
FOR /F "usebackq" %%i IN (`hostname`) DO SET HOSTNAME=%%i
mailsend -d mydomain.com -smtp smtp.mydomain.com
-t admin@mydomain.com -f SafeKit -sub "Stop module %SAFEMODULE% on %HOSTNAME%" -M
"Running poststop" +cc +bc
:nostop
```

⇨ In Linux: **`/opt/safekit/modules/AM/bin/poststop`**

```
#!/bin/sh
# Script called on module stop, stopstart, stopwait
# For logging into SafeKit log use:
# $SAFE/safekit printi | printe "message"
# stdout goes into Application log
echo "Running poststop $*"

if [ "$3" = "stop"]; then
  # send mail only on stop (not on stopstart or stopwait)
  echo "Running poststop" | mail -s " Stop module $SAFEMODULE on `hostname`"
admin@mydomain.com
fi
```

10.7.2 Mail notification on the failover of the module

The user script `transition` can be used to send an e-mail on main local state transitions of the module running on the local server. For instance, it may be useful to know when the mirror module is going `ALONE` (on failover for instance). The script `transition` is not delivered by default and can be created as follow. Replace `AM` by the module name and **<mail …>** by the command for sending e-mail and its arguments.

⇨ In Windows: **c:\safekit\modules\AM\bin\transition.cmd**

```
@echo off
rem Script called on module transitions
rem For logging into module log use:
rem "%SAFE%\safekit" printi | printe "message"

rem stdout goes into Application log
echo "Running transition %*"

rem send an email when transiting from WAIT to ALONE, SECOND to ALONE, PRIM to
ALONE
IF [%1] EQU [WAIT] if [%2] EQU [ALONE] <mail …>
IF [%1] EQU [SECOND] if [%2] EQU [ALONE] <mail …>
IF [%1] EQU [PRIM] if [%2] EQU [ALONE] <mail …>
```

⇨ In Linux: **/opt/safekit/modules/AM/bin/transition**

```
#!/bin/sh
# Script called on module transitions
# For logging into SafeKit log use:
# $SAFE/safekit printi | printe "message"
# stdout goes into Application log
echo "Running transition $*"
# send an email when transiting from WAIT to ALONE, SECOND to ALONE, PRIM to
ALONE
if [ "$1" = "WAIT" -a "$2" = "ALONE" ] ; then <mail …>; fi
if [ "$1" = "SECOND" -a "$2" = "ALONE" ] ; then <mail …> ; fi
if [ "$1" = "PRIM" -a "$2" = "ALONE" ] ; then <mail …> ; fi
```

10.8 SNMP agent

For using the SafeKit SNMP agent `safeagent`, you must:

1. configure it to start on boot, with the command

`safekit boot [snmpon | snmpoff | snmpstatus]`	Controls the automatic start at boot of the `safeagent` service ("on" or "off"; by default, "off")

2. add the corresponding firewall rule

   ✓ In Linux, when using the default `safeagent` configuration (port 3600) and operating system firewall, you can use the command:

   `SAFEBIN/firewallcfg add safeagent`

   ✓ In Windows, when using the operating system firewall, the firewall has already been configured for `safeagent` if you have applied the command:

   `SAFEBIN/firewallcfg add`

3. start it with the command

`safekit safeagent [start | stop | restart | check]`	Controls start/stop of the `safeagent` service that implements the SafeKit SNMP agent.

10.8.1 The SNMP agent configuration

The configuration of the `safeagent` is defined in the self-documented **`SAFE/snmp/conf/snmpd.conf`** file. It is a standard net-snmp configuration file as described in http://net-snmp.sourceforge.net. By default, the service is listening on UDP **`agentaddress`** port `3600` and accepts read request from the public community and write requests from the private community. Read requests are used to get module status and write requests to run actions on the module.

SNMP traps can be sent by SafeKit agent when messages are logged with the I or E levels. The IP address for sending SNMP traps must be set with the line `trapsink SNMPManagerIPaddress` in **`snmpd.conf`**.

You can change the default configuration according to your needs. When you modify `snmpd.conf`, you must manually change the firewall rule and restart the service to load the new configuration with: `safekit safeagent restart`

10.8.2 The SafeKit MIB

The SafeKit MIB is delivered in **`SAFE/snmp/mib/safekit.mib`** (read this file to have the detail of the MIB). Note that the MIB includes a definition of the trap sent by SafeKit.

The SafeKit MIB is accessed with the following identifier (OID, prefix of SafeKit SNMP variables): **`= enterprises.bull.safe.safekit (1.3.6.1.4.1.107.175.10).`**

The SafeKit MIB defines:

➩ The module table: **`skModuleTable`**

The index on the module table is the ID of the application module as returned by the command `safekit module listid`.

Through the MIB, you can read and display the status of an application module on a server (`STOP, WAIT, ALONE, UP, PRIM, SECOND`) or you can take an action on the module (`start`, `stop`, `restart`, `swap`, `stopstart`, `prim`, `second`).

For example, the status of the module with ID 1 is read by an SNMP get to the variable:

```
enterprises.bull.safe.safekit.skModuleTable.skModuleEntry.skModuleCurrentSt
ate.1 = stop (0)
```

Use the `snmp walk` command to check all MIB entries.

- The resource table: **`skResourceTable`**

  Each element defines a resource as for instance the one corresponding to the network interface checker "`intf.192.168.0.0`" and its status (`unknown`, `init`, `up`, `down`).

  Example: SNMP get request to `enterprises.bull.safe.safekit.skResourceTable.skResourceEntry.skResourceName.1.2` means name of resource 2 in application module 1.

- The trap:

  Traps are sent with the OID **`enterprises.bull.safe.safekit.skTrapLogMesg`**. A trap contains the SafeKit last log message (I and E levels) for each module.

10.9 Commands log of the SafeKit server

There is a log of the `safekit` commands ran on the server. It allows auditing the actions performed on the server to help support for instance. The log records all the `safekit` commands that are run and that modify the system such as a module install and configuration, a module start/stop, the `safekit webserver start/stop`, …

Note

Since SafeKit 7.5, this log is stored in the `SAFEVAR/log.db` file in SQLite3 format. In earlier versions, it was stored in the `SAFEVAR/commandlog` text file.

For viewing the commands log:

- run the command `safekit cmdlog`

or

- click on the commands log tab into the web console

Below is the raw extract of this log:

```
| 2021-07-27 14:37:33.205122 |   safekit |   mirror |  6883 | START | config -m
mirror
| 2021-07-27 14:37:33.400513 |   cluster |   mirror |     0 |     I | update
cluster state
| 2021-07-27 14:37:33.405597 |   cluster |   mirror |     0 |     I | module
state change on node centos7-test3
| 2021-07-27 14:37:34.193280 |           |          |  6883 |   END | 0
| 2021-07-27 14:37:34.718292 |   cluster |   mirror |     0 |     I | update
cluster state
| 2021-07-27 14:37:34.722080 |   cluster |   mirror |     0 |     I | module
state change on node centos7-test4
| 2021-07-27 14:37:37.510971 |           |          |  6871 |   END | 0
| 2021-07-27 14:38:05.092924 |   safekit |   mirror |  7017 | START | prim -m
mirror -u web@10.0.0.103
| 2021-07-27 14:38:05.109368 |           |          |  7017 |   END | 0
```

Each field has the following meaning:

- ✓ The 1st field in the log entry is the date and time of the message
- ✓ The next one is the type of the action
- ✓ The next one is the module name when the action is not global
- ✓ The next one is the pid of the process that runs the command. It is used as the identifier of the log entry
- ✓ The next ones are `START` when the command starts and the command’s arguments; or `END` when the command has finished with the return value.

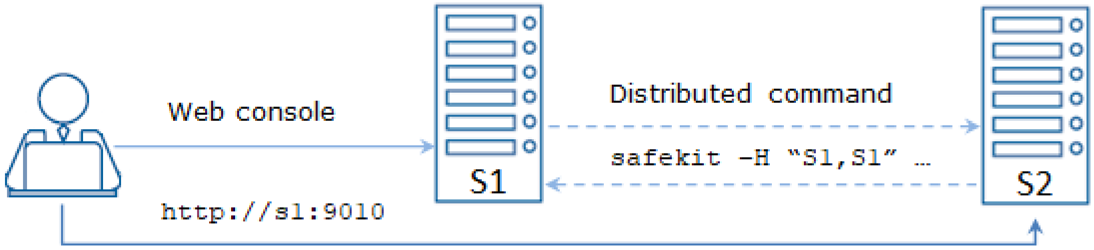
11.Securing the SafeKit web service

11.1 Overview

The SafeKit web service is mainly used by:

- ➾ the web console (see section 3 page 35)
- ➾ the distributed command line interface (see 9.1 page 141)

SafeKit provides different setups for this web service to enhance the security of the SafeKit web console and distributed commands.



Protocol	Authentication	Role management
✓ HTTP	✓ None	✓ Admin
✓ HTTPS	✓ File based	✓ Control
	✓ LDAP/AD	✓ Monitor
	✓ Client certificate	

The most secure setups are based on HTTPS and user authentication.

SafeKit provides a configuration wizard to setup HTTPS, and optionally client certificates, with a “private” certification authority (the SafeKit PKI). This allows SafeKit to be quickly secured without the need for an external PKI (enterprise PKI or commercial PKI) that provides trusted certification authority.

SafeKit offers also optional role management based on 3 roles:

Admin role ● ● ● ●	This role grants all administrative rights by allowing access to the tabs: ● Configuration, ● Control, ● Monitoring and ● Advanced Configuration
Control role ● ●	This role grants control and monitoring rights by allowing access to the tabs: ● Control et ● Monitoring
Monitor role ●	This role only grants monitoring rights by allowing access to the tab: ● Monitoring

11.1.1 Default setup

Since SafeKit 7.5, the default setup is the following:

Setup	Protocol	Authentication Role management
Default	✓ HTTP	✓ File-based authentication (username/password stored in an Apache file) ✓ Initialization with a single user named `admin` with the Admin role To configure, see 11.2.1 page 179

11.1.2 Predefined setups

The predefined setups are as follows:

Setup	Protocol	Authentication Role management
Unsecure	✓ HTTP	✓ No authentication ✓ Same role for all users To configure, see 11.2.2 page 181
File-based	✓ HTTP ✓ HTTPS	✓ username/password stored in an Apache file ✓ Optional role management stored in an Apache file

	To configure HTTPS with: ⇨ the SafeKit PKI, see 11.3.1 page 183 ⇨ your PKI, see 11.3.2 page 188	To configure, see 11.4.1 page 192
LDAP/AD	✓ HTTP ✓ HTTPS To configure HTTPS with: ⇨ the SafeKit PKI, see 11.3.1 page 183 ⇨ your PKI, see 11.3.2 page 188	✓ LDAP/AD authentication ✓ Optional role management To configure, see 11.4.2 page 195
Client certificate	✓ HTTPS To configure HTTPS with: ⇨ the SafeKit PKI, see 11.3.1 page 183 ⇨ your PKI, see 11.3.2 page 188	✓ Client certificate authentication ✓ Integrated role management To configure ⇨ through SafeKit PKI, see 11.4.3 page 197 ⇨ through your PKI, see 11.4.4 page 204

11.2 HTTP setup

By default, after the SafeKit install, the web service is configured for HTTP with file-based authentication that must be initialized.

This default configuration can be extended as described in 11.2.1 page 179.

It can also be replaced by the unsecure setup described in 11.2.2 page 181 or anyone of the predefined setups.

11.2.1 Default setup

Since SafeKit 7.5, the default setup relies on HTTP with file-based authentication. It requires some initialization described below. It is a mandatory step.

This default configuration can be extended:

✓ to add users and assign them a role as described in 11.4.1.1 page 192

✓ to switch to HTTPS with:

⇨ the SafeKit PKI described in 11.3.1 page 183

⇨ your PKI described in 11.3.2 page 188

After the installation of SafeKit, the configuration and restart of the web service is not necessary since this is the default configuration and the web service has been started with it.

If you have changed the default configuration and want to revert to it, see 11.4.1 page 192.

11.2.1.1 Initialization for the web console and distributed command

SafeKit provides a script to get the web console and distributed commands up and running quickly.

In Linux, this script can be automatically called during the install of SafeKit; in Windows, it must be manually executed. In both cases, you will have to give the password value, `pwd` for the `admin` user.

`webservercfg -passwd pwd`	On S1 and S2:
	➾ In Windows, open a PowerShell window as administrator and run (`SAFE=C:\safekit` if `%SYSTEMDRIVE%`=C:)
	`SAFE/private/bin/webservercfg.ps1 -passwd pwd`
	➾ In Linux, open a shell window as root and run (`SAFE=/opt/safekit`)
	`SAFE/private/bin/webservercfg -passwd pwd`
	You must set the same password on all nodes.

Important

The password must be identical on all the nodes of the cluster. Otherwise, web console and distributed commands will fail with authentication errors.

Once this initialization is done on all the cluster nodes:

- you can authenticate in the web console with the name `admin` and the password you provided. The role is Admin by default (unless you change the default behavior by providing the `group.conf` file as described in in 11.4.1.1 page 192)

  On authentication failure in the web console, you may need to reinitialize the `admin` password. For this, run again `webservercfg -passwd pwd` on all nodes.

- you can run distributed commands. It is based on a dedicated user `rcmdadmin` with the Admin role. It is managed in a different, private user file that you do not have to change.

  On authentication failure for distributed commands, you may need to reset `rcmdadmin` password. To reset only this one, without changing the `admin` password, run `webservercfg -rcmdpasswd pwd` on all nodes.

11.2.1.2 Test the web console and distributed command

The setup is complete; you can now test that it is operational.

- Test the web console
  1. Start a browser on the user's workstation
  2. Connect it to the default URL `http://servername:9010` (where `servername` is the name or Ip address of one of the SafeKit nodes)
  3. In the login page, enter as user's name `admin` and as password the one you gave when you initialized it (e.g., `pwd`). Then click on Connect
  4. The loaded page contains all the tabs that correspond to the Admin role by default

- Test the distributed command
  1. Connect on S1 or S2 as administrator/root
  2. Open a system console (PowerShell, shell, …)
  3. Change directory to `SAFE`
  4. Run `safekit -H "*" level`

     that should return the level for all nodes

11.2.2 Unsecure setup based on identical role for all

It is based on the configuration of a single role that is applied to all users without requiring authentication. This solution can only be implemented only in HTTP and is incompatible with user authentication methods.

11.2.2.1 Configure and restart the web service

To configure where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

httpd.conf

On S1 and S2:

- edit `SAFE/web/conf/httpd.conf` file
- comment `usefile` and `useldap`

```
#Define usefile
…
#Define useldap
```

- select the desired role by uncommenting the associated port and commenting all others (Admin role by default)

```
httpadminport (9010)
#httpcontrolport (9010)
#httpmonitorport (9010)
```

	✓ `httpadminport` for Admin role ✓ `httpcontrolport` for Control role ✓ `httpmonitorport` for Monitor role
	On S1 and S2, disable HTTPS if you had configured it: ➾ remove the file `SAFE/web/conf/ssl/httpd.webconsolessl.conf`
	On S1 and S2: ➾ run `safekit webserver restart`

11.2.2.2 Test the web console and distributed command

The setup is complete; you can now test that it is operational.

➾ Test the web console

1. Start a browser on the user’s workstation
2. Connect it to the default URL `http://servername:9010` (where `servername` is the name or Ip address of one of the SafeKit nodes)
3. The loaded page contains only the tabs allowed according to the previously selected port

➾ Test the distributed command

1. Connect on S1 or S2 as administrator/root
2. Open a system console (PowerShell, shell, …)
3. Change directory to `SAFE`
4. Run `safekit -H "*" level`

   that should return the level for all nodes

11.3 HTTPS setup

The HTTPS web service relies on the existence of a set of certificates listed below:

CA	The certificate of the Certification Authority CA used to issue the server certificate for S1 and S2
S1 S2	The server certificate of S1 and S2 used to assert the nodes’ identity

Apply one of the following 2 procedures to configure HTTPS and associated certificates:

- 11.3.1 "HTTPS setup using the SafeKit PKI" page 183

  Go to this section to quickly setup HTTPS with the SafeKit "private" certification authority.

- 11.3.2 "HTTPS setup using an external PKI" page 188

  Go to this section to setup HTTPS with your PKI (enterprise PKI or commercial PKI) that provides trusted certification authority.

At the end of HTTPS setup, you must implement one of the authentication methods described in 11.4 page 192.

11.3.1 HTTPS setup using the SafeKit PKI

Apply the following steps to configure HTTPS with the SafeKit PKI and the associated wizard:

- First, select one node, which belongs to the SafeKit cluster, to apply the first configuration. The selected node will be hereafter called the `first server` (or `CA server`). This server will also act as the Certificate Authority server for the other SafeKit cluster nodes. Apply steps from 11.3.1.1 to 11.3.1.4 to setup HTTPS on the first server, S1 for example.
- The other cluster nodes are called `additional server` (or `non-CA server`). For all additional servers apply steps from 11.3.1.5 to 11.3.1.8 to setup HTTPS on them. S2 in the example.
- Apply step 11.3.1.9 for stopping the CA web service on all additional servers. S2 in the example. If you want to setup client certificates as authentication method, before stopping the CA web service on the first server, S1 in the example, apply the procedure described in 11.4.3 page 197.

Important

Verify that the system clock is set to the current date and time on all SafeKit nodes and workstations that will run the HTTPS SafeKit web console. Certificates are timestamped, and a time difference between systems may have an impact on certificate validity.

11.3.1.1 Start the CA web service on the first server

On the server:

1. Log as administrator/root and open a command shell window
2. Change to the directory `SAFE/web/bin`
3. Run the command `./startcaserv`

   When prompted, enter a password to protect the access to this service for the `CA_admin` user (for instance, `PasW0rD)`. This command starts the `safecaserv` service.

Note

Remember this password since it will be required to connect to this service in next steps.

The CA web service running on the first server is also accessed by the additional servers and SafeKit web console clients for downloading certificate signatures and certificates.

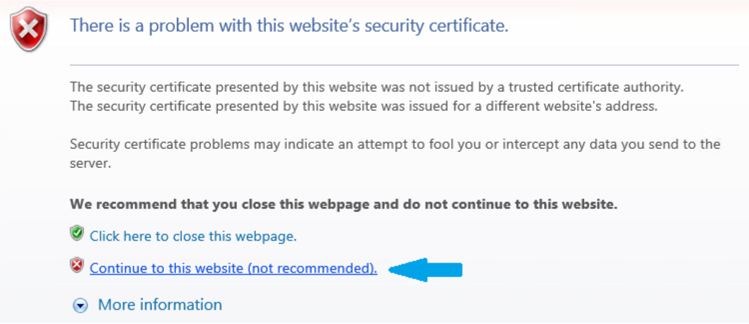
Important

Since the service listens to TCP port 9001, make sure TCP port 9001 is not used, and is allowed in the firewall configuration. On Linux, the TCP 9001 port is automatically opened in local firewall by the `startcaserv` command. In Windows, the `firewallcfg` command opens `safecaserv` service communications.

11.3.1.2 Start the HTTPS configuration wizard on the first server

Launch the HTTPS configuration wizard by starting a local web browser on the server and connecting it to https://localhost:9001.

The certificate associated with the local CA web service is self-signed; therefore, the browser will display a security warning saying the certificate is invalid. This is expected, and you must click "Continue to this website (not recommended)" to continue



There is a problem with this website's security certificate.

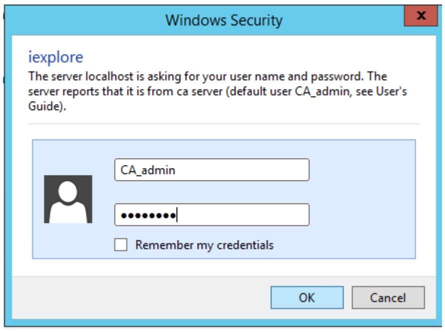
The security certificate presented by this website was not issued by a trusted certificate authority.
The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.
Continue to this website (not recommended).
More information

At the login prompt, enter `CA_admin` as username, and the password you specified when you started the CA web service (for instance, `PasW0rD`)



Windows Security

iexplore

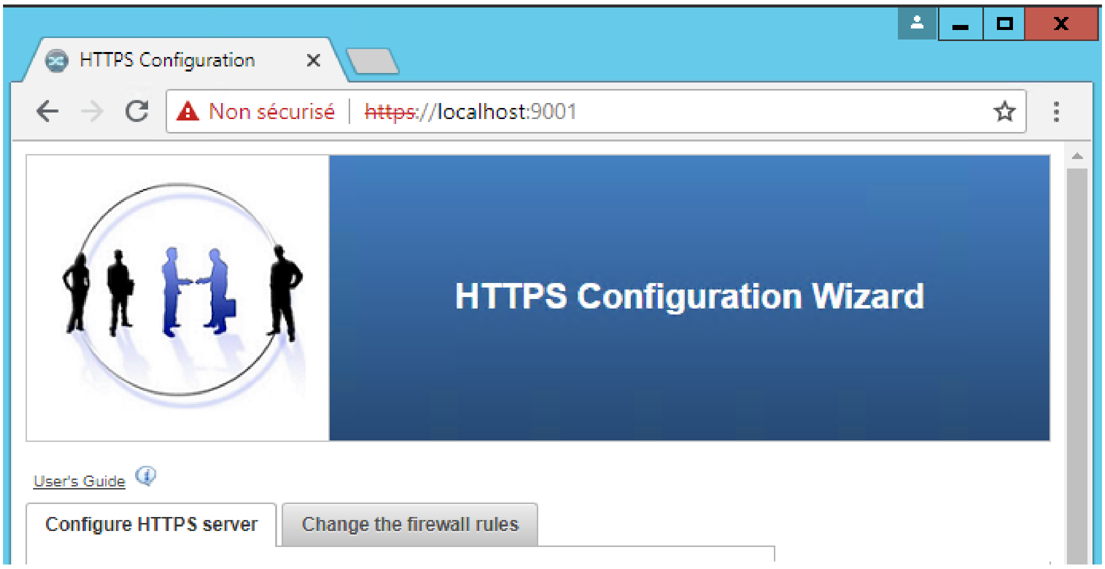
The server localhost is asking for your user name and password. The server reports that it is from ca server (default user CA_admin, see User's Guide).

CA_admin

••••••••

□ Remember my credentials

At this step, the HTTPS configuration wizard is opened.

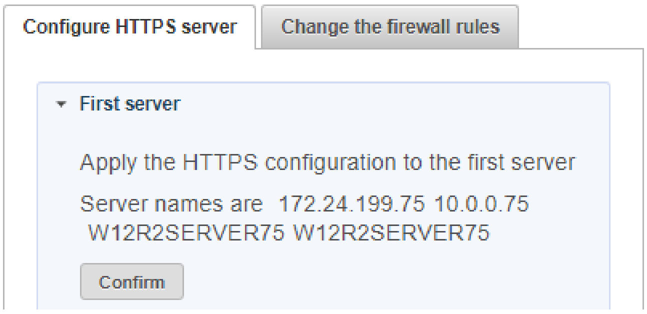


- ✓ Some advanced configuration methods are present into the … panel not described in this document
- ✓ The commands log panel at the bottom of the wizard displays the output of actions that have been executed

11.3.1.3 Configure HTTPS on the first server

This step setup the first server for HTTPS:

- ➾ In HTTPS configuration wizard
- ➾ Go to Configure HTTPS server tab
- ➾ Open First server panel



Apply the HTTPS configuration to the first server

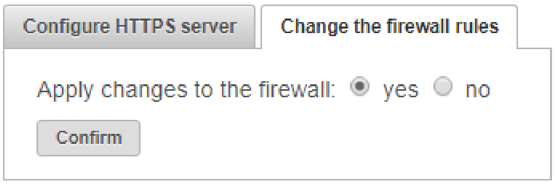
Server names are 172.24.199.75 10.0.0.75 W12R2SERVER75 W12R2SERVER75

- ➾ Click on the Confirm button

When the processing is completed, the Certification Authority is initialized and the certificates necessary to run the SafeKit web service (`safewebserver` service) in HTTPS mode are locally installed. Moreover, this service has been reconfigured for HTTPS and restarted (by applying the procedure described in section 10.6.4 page 170).

11.3.1.4 Change the firewall rules on the first server

- In HTTPS configuration wizard
- Go to Change the firewall rules tab



Apply changes to the firewall: ● yes ○ no

- Select yes to automatically change rules

  It consists in running the `firewallcfg` script, that applies default rules for SafeKit to the operating system default firewall (in Windows, `Microsoft Windows Firewall` ; in Linux, `firewalld` or `iptables`).

- Select no to not change rules

  Choose this option if you want to configure the firewall yourself or if you use a different firewall than the system one. For the list of SafeKit processes and ports, see 10.3 page 156.

- Click on the Confirm button to apply your selection

11.3.1.5 Start the CA web service on additional server

Once the first server is configured, you must configure all the additional servers. Apply the same procedure as the one described in 11.3.1.1 page 183, for starting the CA web service (`safecaserv` service) on the other server(s).

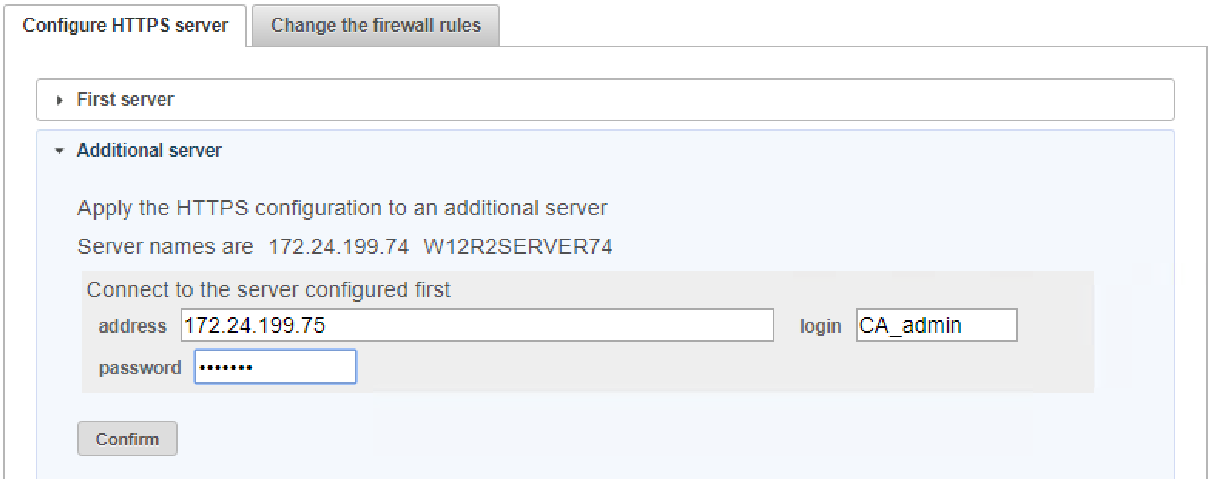
11.3.1.6 Start the HTTPS configuration wizard on additional server

Apply the same procedure as the one described in 11.3.1.2 page 184, for launching the configuration wizard on the additional server(s).

11.3.1.7 Configure HTTPS on additional server

This step enables HTTPS on a server different from the first server:

- In HTTPS configuration wizard that is running on the additional server
- Go to Configure HTTPS server tab
- Open Additional server panel



Apply the HTTPS configuration to an additional server

Server names are 172.24.199.74 W12R2SERVER74

Connect to the server configured first

address: 172.24.199.75

login: CA_admin

password: •••••••

- ➾ Fill in the IP address of the first server
- ➾ Fill in the password you specified when you started the CA web service on the first server (for instance, `PasW0rD`)
- ➾ Click on the Confirm button

When the processing is completed, the certificates necessary to run the SafeKit web service (`safewebserver` service) in HTTPS mode are locally installed. Moreover, this service has been reconfigured and restarted for HTTPS (by applying the procedure described in 10.6.4 page 170).

11.3.1.8 Change the firewall rules on all additional servers

Apply the same procedure as the one described in 11.3.1.4 page 186, for configuring the firewall on additional server.

11.3.1.9 Stop the CA web service on the first server and additional servers

If you want to setup client certificates as authentication methods, before stopping the CA web service on the first server, apply the procedure described in 11.4.3 page 197.

Once all SafeKit nodes and clients have been configured, bring the CA web service (`safecaserv` service) offline on all servers, to limit the risk of accidental or malicious access to the configuration wizard.

To stop the SafeKit CA web service:

1. Log as administrator/root and open a command shell window
2. Change to the directory `SAFE/web/bin`
3. Run the command `./stopcaserv`

Note

On Windows, this command also removes the service entry to prevent any accidental start of the service afterwards. On Linux, the 9001 port is automatically closed on local firewall.

11.3.2 HTTPS setup using an external PKI

Apply steps below to setup HTTPS with your trusted certification authority (your enterprise PKI or commercial PKI).

11.3.2.1 Get and install server certificates

11.3.2.1.1 Get certificate files

You must get server certificates from your PKI with the expected format.

⚠ Be aware that you must provide all names and/or IP addresses, for S1 and S2, that are used for HTTPS connections. These ones must also be included into the SafeKit cluster configuration file. See the example in 11.3.2.1.3 page 189.

CA	The certificate of the Certification Authority CA used to issue the server certificates
S1	The server certificate to assert the S1 identity.
S2	The server certificate to assert the S2 identity.
`server1.crt` `server2.crt`	⇨ X509 certificate file in PEM format The subfield `CN` (`Common Name`) into the `subject` field, or the `Subject Alternative Name` field of the certificate, must contain : ✓ localhost and 127.0.0.1 ✓ S1 name(s) and/or IP address(es) for `server1.crt` ✓ S2 names and/or IP address(es) for `server2.crt` ⚠ With SafeKit <= 7.5.2.9, the server’s name must be included.
`server1.key` `server2.key`	⇨ The private, *unencrypted* key corresponding to the certificates `server1.crt` and `server2.crt`

11.3.2.1.2 Install files in SafeKit

Install the certificates as follow (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

S1 server1.crt server1.key	On S1: ⇨ copy `server1.crt` to `SAFE/web/conf/server.crt` ⇨ copy `server1.key` to `SAFE/web/conf/server.key`
S2 server2.crt server2.key	On S2: ⇨ copy `server2.crt` to `SAFE/web/conf/server.crt` ⇨ copy `server2.key` to `SAFE/web/conf/server.key`

You can check the installed certificates with:

```
cd SAFE/web/bin
checkcert -t server
```

It returns a failure if an error is detected.

You can check that the certificate contains some DNS name or IP address with:

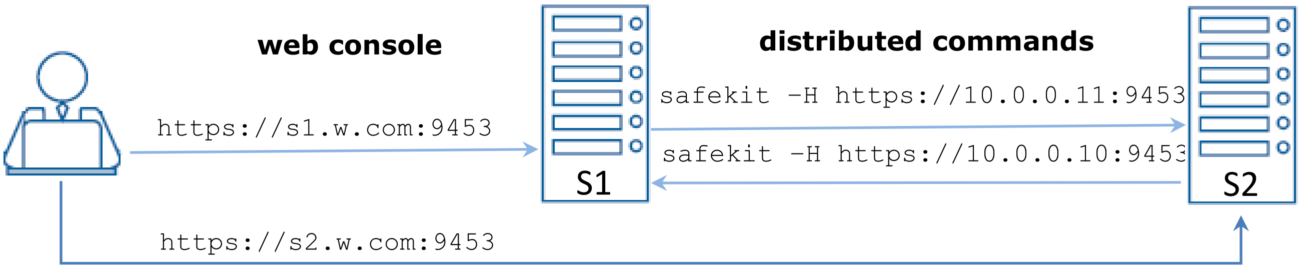
```
checkcert -h ”DNS name value”
checkcert -i ”Numeric IP address value”
```

Check that it also contains localhost and 127.0.0.1:

```
checkcert -h ”localhost”
checkcert -i ”127.0.0.1”
```

11.3.2.1.3 Example

Consider the following architecture:



The corresponding SafeKit cluster configuration file, `SAFEVAR/cluster/cluster.xml` must contain these values into `addr` field:

```
<?xml version="1.0" encoding="UTF-8"?>
<cluster>
<lans>
  <lan name="default" console="on" framework="on">
    <node name="s1" addr="10.0.0.10"/>
    <node name="s2" addr="10.0.0.11"/>
  </lan>
  <lan name="console" console="on" framework="off">
    <node name="s1" addr="s1.w.com"/>
    <node name="s2" addr="s2.w.com"/>
  </lan>
</lans>
</cluster>
```

The server certificates must contain the same values (DNS names and/or IP addresses) as those in the cluster configuration. If not, the SafeKit web console and distributed commands will not work properly.

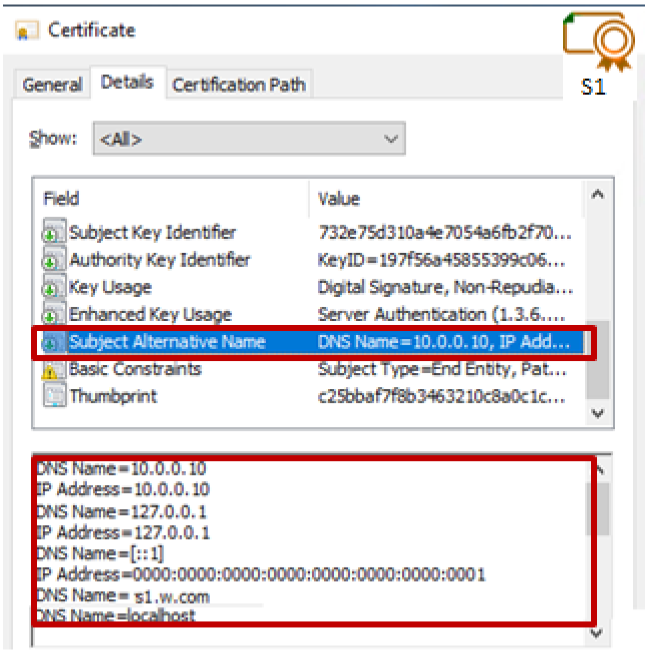
To check that the certificate file is correct:

1. Copy the `.crt` (or `.cer`) file on a Windows workstation
2. Double click on this file to open it with Crypto Shell Extensions
3. Click on the Details tab
4. Verify the `Subject Alternative Name` field

If you prefer the command line interface, you can run on each the SafeKit node:

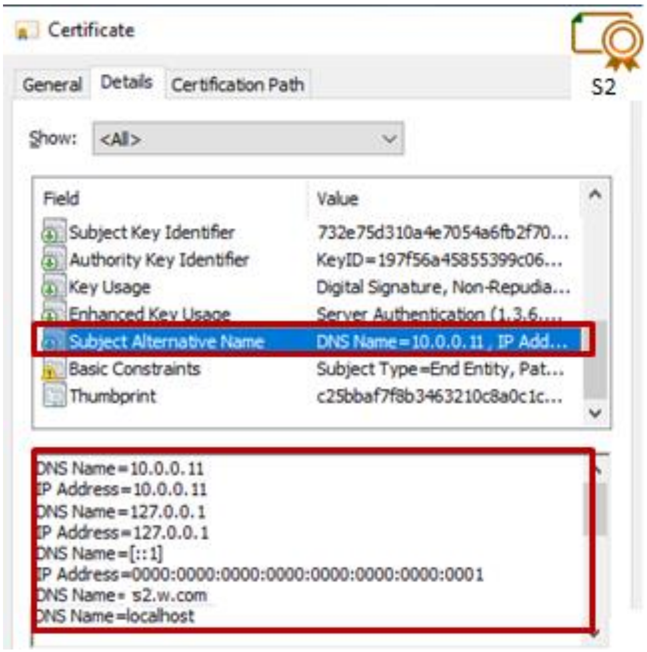
`SAFE/web/bin/openssl.exe x509 -text -noout -in SAFE/web/conf/server.crt`

and look for the value after `Subject Alternative Name`



Field	Value
Subject Key Identifier	732e75d310a4e7054a6fb2f70...
Authority Key Identifier	KeyID=197f56a45855399c06...
Key Usage	Digital Signature, Non-Repudia...
Enhanced Key Usage	Server Authentication (1.3.6....
Subject Alternative Name	DNS Name=10.0.0.10, IP Add...
Basic Constraints	Subject Type=End Entity, Pat...
Thumbprint	c25bbaf7f8b3463210c8a0c1c...

```
DNS Name=10.0.0.10
IP Address=10.0.0.10
DNS Name=127.0.0.1
IP Address=127.0.0.1
DNS Name=[::1]
IP Address=0000:0000:0000:0000:0000:0000:0000:0001
DNS Name= s1.w.com
DNS Name=localhost
```



Field	Value
Subject Key Identifier	732e75d310a4e7054a6fb2f70...
Authority Key Identifier	KeyID=197f56a45855399c06...
Key Usage	Digital Signature, Non-Repudia...
Enhanced Key Usage	Server Authentication (1.3.6....
Subject Alternative Name	DNS Name=10.0.0.11, IP Add...
Basic Constraints	Subject Type=End Entity, Pat...
Thumbprint	c25bbaf7f8b3463210c8a0c1c...

```
DNS Name=10.0.0.11
IP Address=10.0.0.11
DNS Name=127.0.0.1
IP Address=127.0.0.1
DNS Name=[::1]
IP Address=0000:0000:0000:0000:0000:0000:0000:0001
DNS Name= s2.w.com
DNS Name=localhost
```

⇨ localhost and 127.0.0.1 must be present

⇨ with SafeKit <= 7.5.2.9, the server's name must be present

11.3.2.2 Get and install the CA certificate

11.3.2.2.1 Get certificate file

You must get these certificates from your PKI with the expected format.

CA `cacert.crt`	The Certification Authority CA certificate used to issue the server certificates. ⇨ X509 certificate file in PEM format The chain of certificates for the root and intermediates CA	S1 S2 Server certificates for S1 and S2

If you have trouble retrieving this file from your PKI, you can build it using the procedure described in 7.18 page 128.

11.3.2.2.2 Install file in SafeKit

Install certificates files as follow (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

CA `cacert.crt`	On S1 and S2: ⇨ copy `cacert.crt` to `SAFE/web/conf/cacert.crt`

You can check the installed certificates with:

```
cd SAFE/web/bin
checkcert -t CA
```

It returns a failure if an error is detected.

You must also check that the `cacert.crt` contains the chain of certificates for the root and intermediates Certification Authorities.

11.3.2.3 Configure and restart the web service

To enable HTTPS (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

httpd.webconsolessl.conf	On S1 and S2: ⇨ copy `SAFE/web/conf/httpd.webconsolessl.conf` to `SAFE/web/conf/ssl/httpd.webconsolessl.conf`
	On S1 and S2: ⇨ run `safekit webserver restart`

11.3.2.4 Change the firewall rules

You can run the `firewallcfg` script to change the firewall rules. It set SafeKit rules into the operating system default firewall (in Windows, `Microsoft Windows Firewall` ; in Linux, `firewalld` or `iptables`).

Firewall	On S1 and S2: ⇨ run `SAFEBIN/firewallcfg add`

Don’t run this command if you want to configure the firewall yourself or if you use a different firewall than the system one. For the list of SafeKit processes and ports, see 10.3 page 156.

11.4 User authentication setup

Setup one of the following user authentication methods:

- ⇨ 11.4.1 “File-based authentication setup” page 192
- ⇨ 11.4.2 “LDAP/AD authentication setup” page 195
- ⇨ 11.4.3 “Client certificate authentication setup using the SafeKit PKI” page 197
- ⇨ 11.4.4 “Client certificates authentication setup using an external PKI” page 204

At the end of this setup, you can start using the secure SafeKit web console.

11.4.1 File-based authentication setup

File-based authentication setup can be applied in HTTP or HTTPS. It relies on the following files:

user.conf	User file configuration that defines authorized users
Admin Control Monitor group.conf	Optional file to restrict the user’s role. If the `group.conf` file is not present, all authenticated users will have the Admin role.

11.4.1.1 Manage users and groups

The users and groups must be identical on S1 and S2, as well as passwords. It is defined by the files `user.conf` and `group.conf` into `SAFE/web/conf` directory (`SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux).

Note

During the default setup initialization, described in 11.2.1 page 179, the user named `admin` has been created and thus is present into `user.conf`. You can decide to remove this user if you create others.

➾ Create a new user

Users are created with the `SAFE/web/bin/htpasswd` command.

For instance, to add the new user `manager` and set its password `managerpassword`, run:

```
SAFE/web/bin/htpasswd -b SAFE/web/conf/user.conf manager managerpassword
```

The new user is inserted into `SAFE/web/conf/user.conf` the file.

user.conf

```
admin:$2y$05$oPquL6Z2Y78QcXpHIako.O58Z6lWfa5A86XD.eCbEnbRcguJln9Ce
manager:$apr1$U2GLivF5$x39WKmSpq6BGmLybESgNV1
operator1:$apr1$DetdwaZz$hy5pQzpUlPny3qsXrIS/z1
operator2:$apr1$ICiZv2ru$wRkc3BclBhXzc/4llofoc1
```

➾ Assign the role of the new user

By default, all users have the Admin role. If you want to assign different roles to different users, you must create the `SAFE/web/conf/group.conf` file and assign user's role. The group file can contain the 3 groups Admin, Control, Monitor. Users in these groups will have the corresponding roles.

Note

Each line of the group file must contain the group name followed by a colon, followed by the member users name separated by spaces. See the example above.

For instance, assign the Control role to the new user `manager`:

group.conf

```
Admin : admin
Control : manager
Monitor : operator1 operator2
```

Note

If you enable the role management, you must insert the user `admin` into `group.conf`. Otherwise, this user will no longer be operational.

➾ Delete a user, …

Use `htpasswd -?` for all user management commands (add/delete, ...).

11.4.1.2 Install files

Install the files as follow (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

user.conf	On S1 and S2: ⇨ copy `user.conf` to `SAFE/web/conf/user.conf`
Admin Control Monitor group.conf	On S1 and S2 if groups are set: ⇨ copy `group.conf` to `SAFE/web/conf/group.conf`

These files must be identical on all nodes.

11.4.1.3 Configure and restart the web service

To configure the file-based authentication (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

httpd.conf	On S1 and S2:
	⇨ edit `SAFE/web/conf/httpd.conf` file
	⇨ uncomment `usefile`
	`Define usefile`
	⇨ verify that `useldap` is commented
	`# Define useldap`
	⇨ verify that only `httpadminport` is defined
	`httpadminport (9010)`
	`#httpcontrolport (9010)`
	`#httpmonitorport (9010)`

	On S1 and S2: ⇨ run `safekit webserver restart`

Since SafeKit 7.5, this is the default content of `httpd.conf`.

11.4.1.4 Test the web console and distributed command

The setup is complete; you can now test that it is operational.

⇨ Test the web console

1. Start a browser on the user's workstation
2. Connect it to the default URL `http://servername:9010` (where `servername` is the name or Ip address of one of the SafeKit nodes). If HTTPS is configured, there is an automatic redirection to `https://servername:9453`

3. In the login page, fill in the user's name and password then click on Connect

   With the SafeKit 7.5 default configuration, you can log-in with the user `admin` by giving the password you assigned during initialization.

4. The loaded page contains only the tabs allowed according to the user's role. If the groups have not been defined, all users have the Admin role.

⇨ Test the distributed command

1. Connect on S1 or S2 as administrator/root
2. Open a system console (PowerShell, shell, …)
3. Change directory to `SAFE`
4. Run `safekit -H "*" level`

   that should return the level for all nodes

11.4.2 LDAP/AD authentication setup

LDAP/AD authentication setup can be applied in HTTP or HTTPS. It requires:

👤	LDAP/Active Directory account configuration used to assert the user identity
👥 ADMIN 👥 CONTROL 👥 MONITOR	Optional LDAP/Active Directory group configuration to restrict the user's role. When groups are not defined, all authenticated users have the Admin role.

Note

On some Linux distributions (such as RedHat 8 and CentOS 8), the web server start fails when it is configured with LDAP/AD authentication. In this case, apply the solution described in SK-0092.

Apply the steps described below after verifying that S1 and S2 can connect to the LDAP controller domain port (default is 389).

11.4.2.1 Manage users and groups

If necessary, ask your LDAP administrator to create users of the SafeKit web console.

If you want to define user's role, ask your LDAP administrator to create groups for Admin, Control, Monitor roles and assign users to groups. When groups are not defined, all users will have the Admin role.

11.4.2.2 Configure and restart the web service

To configure the LDAP/AD authentication (where `SAFE=C:\safekit` in Windows if `%SYSTEMDRIVE%`=C: ; and `SAFE=/opt/safekit` in Linux):

On S1 and S2:

Initialize the authentication for the distributed command. This may have already been done if you initialized the default configuration after SafeKit installation. Otherwise:

➩ Run `webservercfg -rcmdpasswd pwd`

where `pwd` is the password for the private user `rcmdadmin`. You don’t need to memorize it.

httpd.conf

On S1 and S2:

➩ edit `SAFE/web/conf/httpd.conf` file

➩ comment `usefile`

```
#Define usefile
```

➩ uncomment `useldap`

```
Define useldap
```

➩ verify that only `httpadminport` is defined

```
httpadminport (9010)
#httpcontrolport (9010)
#httpmonitorport (9010)
```

➩ uncomment the following lines and replace bold values according to your LDAP/AD service configuration:

```
Define binddn "CN=bindCN,OU=bindOU1,OU=bindOU2,DC=domain,DC=fq,DC=dn"
Define bindpwd "Password0"
Define searchurl "ldap://ldaporad.fq.dn:389/OU=searchou, DC=domain, DC=fq,
DC=dn?sAMAccountName, memberOf?sub?(objectClass=*)"
```

✓ the `binddn` and `bindpwd` variables must contain the credentials of an account with search rights on the directory

✓ the `searchurl` variable defines the RFC2255 search URL to authenticate the user

Note

`CN`: common name

`OU`: organization unit

`DC`: domain component (one field for each part of the FQDN)

If the group configuration is not enabled, all authenticated users will have the Admin role.

On S1 and S2

To enable group management:

➩ edit `SAFE/web/conf/httpd.conf` file

- uncomment the following lines and replace bold values according to your LDAP/AD service configuration:

```
Define admingroup
"CN=Group1CN,OU=Group1OU1,OU=Group1OU2,DC=domain,DC=fq,DC=dn"
Define controlgroup
"CN=Group2CN,OU=Group2OU1,OU=Group2OU2,DC=domain,DC=fq,DC=dn"
Define monitorgroup
"CN=Group3CN,OU=Group3OU1,OU=Group3OU2,DC=domain,DC=fq,DC=dn"
```

Users set into the LDAP/AD groups associated to `admingroup`, `controlgroup` and `monitorgroup`, will respectively have Admin, Control and Monitor roles.

For more sophisticated authentication, read Apache web service documentation (see http://httpd.apache.org).

On S1 and S2:

- run `safekit webserver restart`

11.4.2.3 Test the web console and distributed command

The setup is complete; you can now test that it is operational.

- Test the web console

1. Start a browser on the user's workstation
2. Connect it to the default URL `http://servername:9010` (where `servername` is the name or Ip address of one of the SafeKit nodes). If HTTPS is configured, there is an automatic redirection to `https://servername:9453`
3. In the login page, fill in the user's name and password then click on Connect
5. The loaded page contains only the tabs allowed according to the user's role. If the groups have not been defined, all users have the Admin role.

- Test the distributed command

1. Connect on S1 or S2 as administrator/root
2. Open a system console (PowerShell, shell, …)
3. Change directory to `SAFE`
4. Run `safekit -H "*" level`

   that should return the level for all nodes

11.4.3 Client certificate authentication setup using the SafeKit PKI

Once the HTTPS setup has been completed, as described in 11.3.1 page 183, you can go on client certificates authentication setup:

1. the web service configuration must be modified to enable client certificates authentication as described in 11.4.3.1

2. the Client certificate configuration wizard helps to create, download, and import client certificates on the user’s workstation. To do this, apply sections 11.4.3.2 to 11.4.3.5 on each workstation of the users concerned

Important

Verify that the system clock is set to the current date and time for all clients. Certificates are timestamped, and a time difference between systems may have an impact on certificate validity.

The configuration wizards associated with the SafeKit PKI manage the creation of all the certificates required for setting up the client certificate authentication for the web console and distributed commands:

CLCA	The certificate of the Certification Authority CLCA used to issue the client certificates
ADMIN CONTROL MONITOR	The client certificates used to assert the user identity and its role in the console
ADMIN1 ADMIN2	The client certificates used to assert the administrator identity on S1 and S2 for distributed commands
PROXY1 PROXY2	The client certificates used to assert the administrator identity on S1 and S2 for the console in proxy mode. They are built from `admin1` and `admin2` certificates.

This implementation corresponds to the one supported since SafeKit 7.4. It has been slightly simplified since SafeKit 7.5 but only for the configuration with an external PKI.

11.4.3.1 Configure and restart the web service

To enable client certificates authentication (where `SAFE=C:\safekit` in Windows if `%SYSTEMDRIVE%`=C: ; and `SAFE=/opt/safekit` in Linux):

httpd.conf

On S1 and S2:

⇨ edit `SAFE/web/conf/httpd.conf` file

⇨ comment `usefile` and `useldap`

```
# Define useldap
…
# Define usefile
```

- verify that only httpadminport is defined

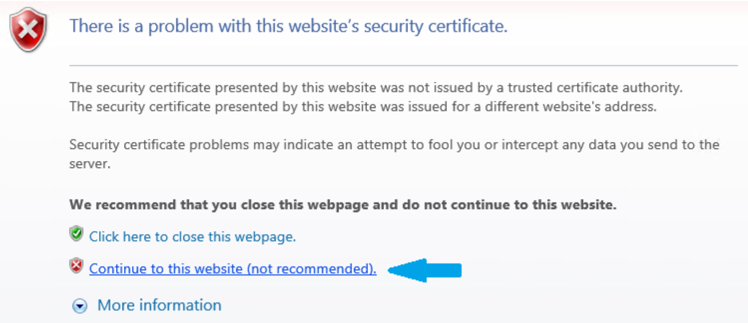
```
httpadminport (9010)
#httpcontrolport (9010)
#httpmonitorport (9010)
```

On S1 and S2:

- run safekit webserver restart

11.4.3.2 Start the Client certificate configuration wizard

- Log on the workstation of the user that will access to the console
- Connect a browser to https://firstserver:9001/adduser.html where firstserver is the IP address of the first server configured for HTTPS
- The certificate associated with the CA web service is self-signed; therefore, the browser will display a security warning saying the certificate is invalid. This is expected, and you must click through the warning to continue.



There is a problem with this website's security certificate.

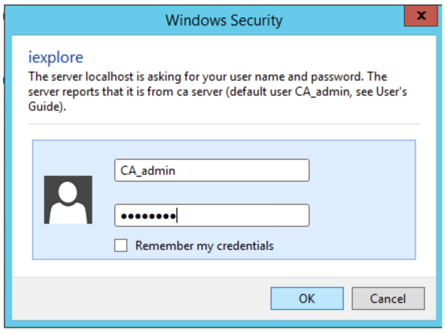
The security certificate presented by this website was not issued by a trusted certificate authority.
The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.
Continue to this website (not recommended).
More information

- At the login prompt, enter CA_admin as username, and the password you specified when you started the CA web service on the CA server (for instance, PasW0rD)

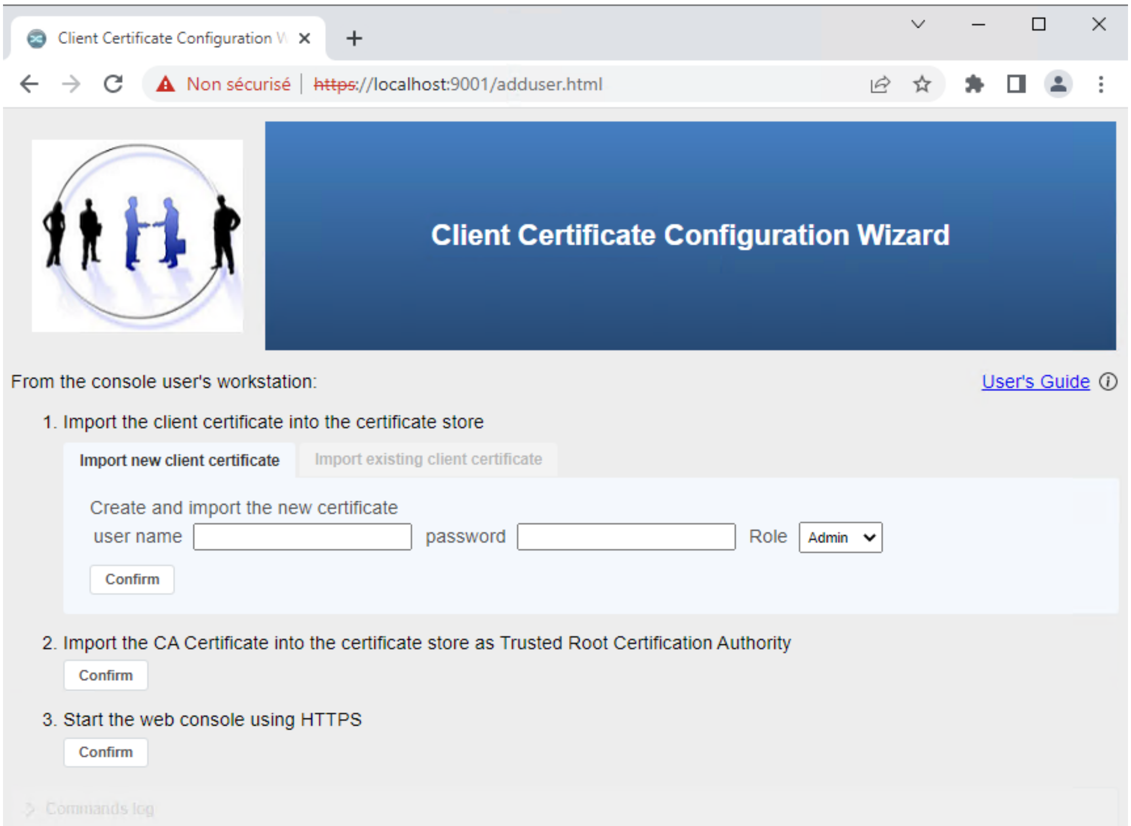


The server localhost is asking for your user name and password. The server reports that it is from ca server (default user CA_admin, see User's Guide).

CA_admin

••••••••

At this step, the Client certificate configuration wizard is opened.


Client Certificate Configuration W
Non sécurisé
https://localhost:9001/adduser.html
Client Certificate Configuration Wizard
User's Guide
Import new client certificate
Import existing client certificate
Confirm
Commands log

From the console user's workstation:

1. Import the client certificate into the certificate store

Create and import the new certificate

user name password Role Admin

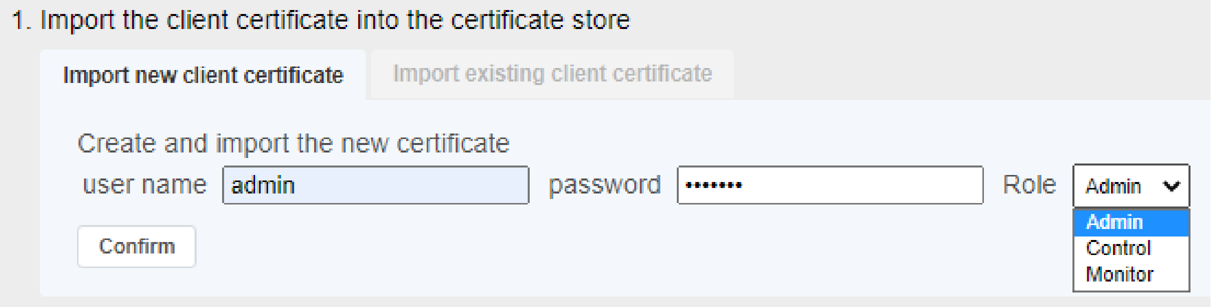
2. Import the CA Certificate into the certificate store as Trusted Root Certification Authority

3. Start the web console using HTTPS

11.4.3.3 Create and/or download the client certificates

Create a new client certificate for the user if it does not already exist.

➪ Create a new client certificate


Import new client certificate
Import existing client certificate
Confirm
Admin
Control
Monitor

1. Import the client certificate into the certificate store

Create and import the new certificate

user name admin password ••••••• Role Admin

1. Fill in the user name, password and role fields of the form. Please note that the username must be unique.
2. Click on Confirm

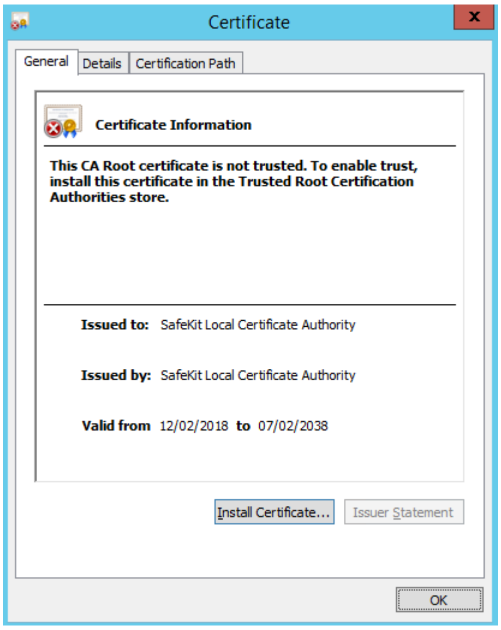
   After the form is processed, the resulting client certificate (the `user_Admin_administrator.p12` file) is downloaded

Once the client certificate is downloaded (the new one or the existing one), import it into the user’s workstation certificate store. Web console access is denied until you import the client certificate.

11.4.3.4 Import the client certificate into the personal certificate store

The procedure depends on the browser and/or the operating system used. The following describes the installation in Windows.

- Click on the downloaded `.p12` file (for instance `user_Admin_administrator.p12)` for opening the certificate window. Then click on Install Certificate button.



Certificate

Certificate Information

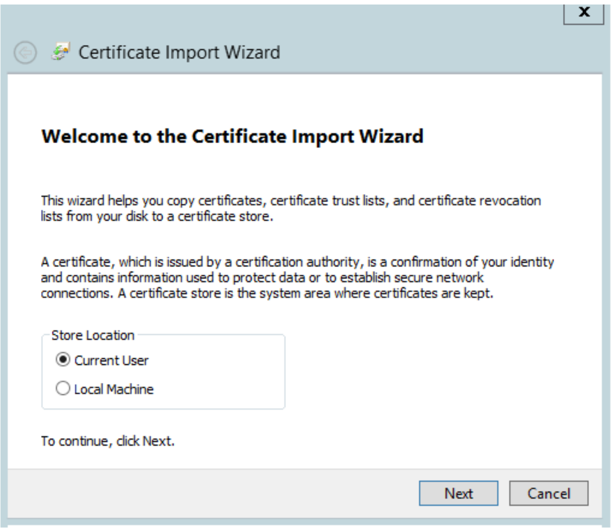
This CA Root certificate is not trusted. To enable trust, install this certificate in the Trusted Root Certification Authorities store.

Issued to: SafeKit Local Certificate Authority

Issued by: SafeKit Local Certificate Authority

Valid from 12/02/2018 to 07/02/2038

- It opens the Certificate Import Wizard. Select Current User and click on the Next button. Go on until the wizard requires the password that protects the certificate.



Certificate Import Wizard

Welcome to the Certificate Import Wizard

This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.

A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

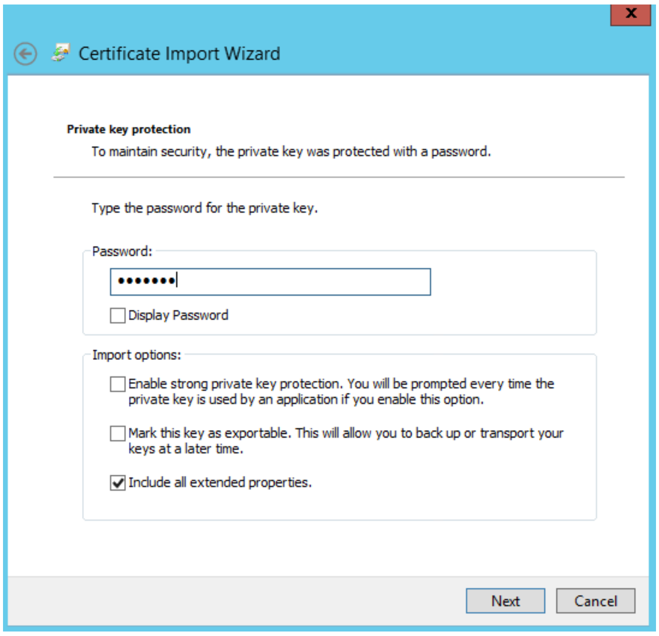
Store Location

● Current User

○ Local Machine

To continue, click Next.

- Enter the password when required. The password to use is the one set during client certificate creation described above



Certificate Import Wizard

Private key protection

To maintain security, the private key was protected with a password.

Type the password for the private key.

Password: •••••••

□ Display Password

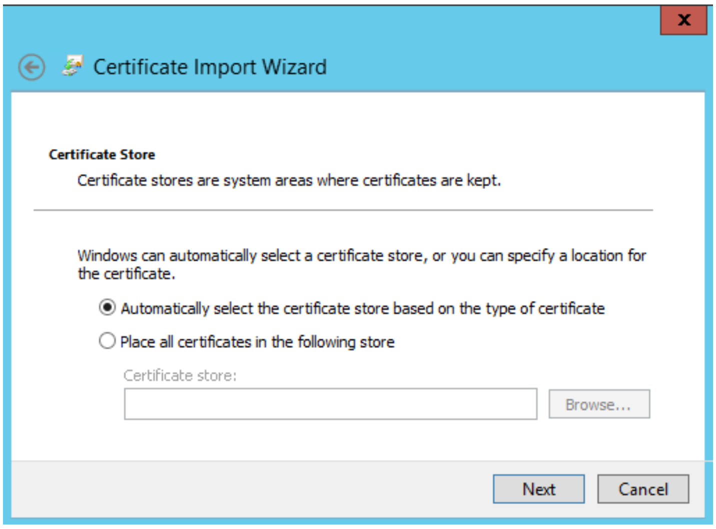
Import options:

□ Enable strong private key protection. You will be prompted every time the private key is used by an application if you enable this option.

□ Mark this key as exportable. This will allow you to back up or transport your keys at a later time.

☑ Include all extended properties.

- Let the wizard automatically select the certificate store that is the Personal store.



Certificate Import Wizard

Certificate Store

Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

● Automatically select the certificate store based on the type of certificate

○ Place all certificates in the following store

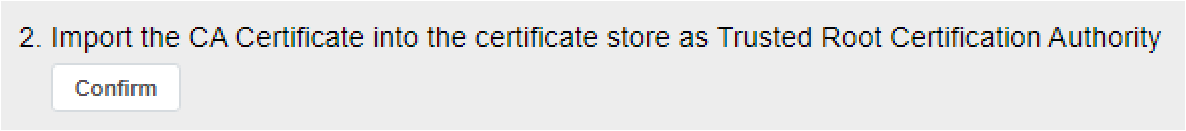
Certificate store:

- Then complete the certificate import.

11.4.3.5 Import the CA certificate as trusted root certification authority

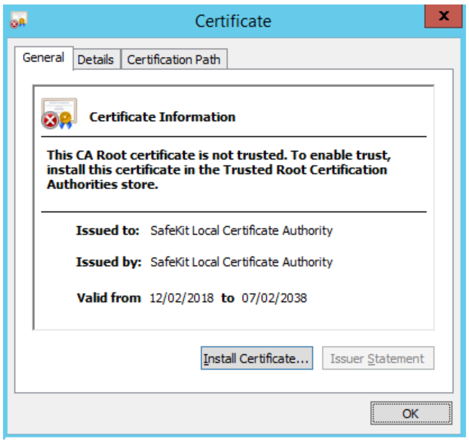
The browser will issue security warnings when you connect to the SafeKit web console unless you import the CA certificate. The procedure depends on the browser and the operating system used. The following describes the installation in Windows.

- Click on Confirm to download the CA certificate



2. Import the CA Certificate into the certificate store as Trusted Root Certification Authority

- Click on the downloaded `cacert.crt` file for opening the certificate window. Then click on Install Certificate button



Certificate Information

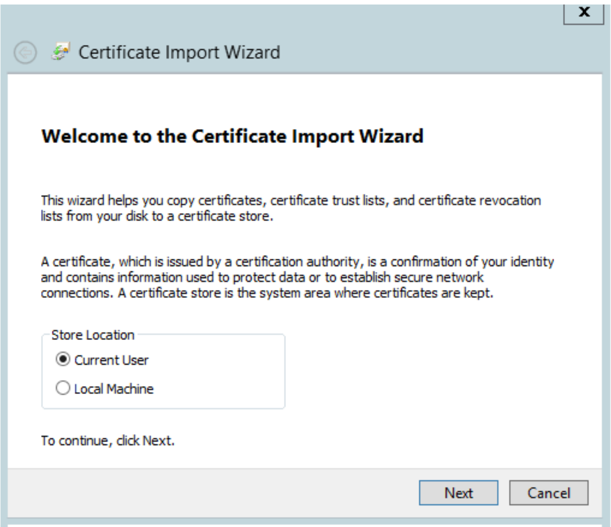
This CA Root certificate is not trusted. To enable trust, install this certificate in the Trusted Root Certification Authorities store.

Issued to: SafeKit Local Certificate Authority

Issued by: SafeKit Local Certificate Authority

Valid from 12/02/2018 **to** 07/02/2038

- It opens the Certificate Import Wizard. Select Current User and click on the Next button



Welcome to the Certificate Import Wizard

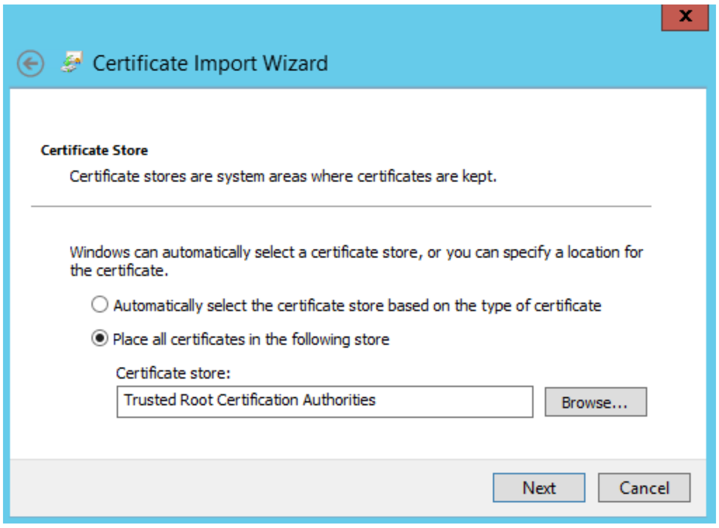
This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.

A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

Store Location
● Current User
○ Local Machine

To continue, click Next.

- Browse stores to select the Trusted Root Certification Authorities store. Then click on Next button



Certificate Store

Certificate stores are system areas where certificates are kept.

Windows can automatically select a certificate store, or you can specify a location for the certificate.

○ Automatically select the certificate store based on the type of certificate
● Place all certificates in the following store

Certificate store: Trusted Root Certification Authorities

- Then complete the certificate import.

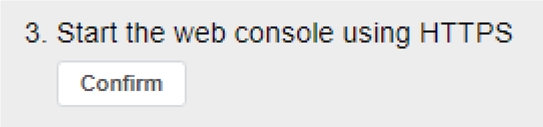
11.4.3.6 Test the web console and distributed command

The setup is complete; you can now test that it is operational.

⇨ Test the web console

Once certificates are imported on the user workstation, the secure SafeKit web console can be used.

1. Click on the Confirm button



3. Start the web console using HTTPS

or

1. Start a browser on the user’s workstation
2. Connect it to the default URL `http://servername:9010` (where `servername` is the name or Ip address of one of the SafeKit nodes. Since HTTPS is configured, there is an automatic redirection to `https://servername:9453`
3. Depending on the browser and the server IP address, you may sometimes need to select the client certificate to use (the friendly name is displayed)
4. The loaded page contains only the tabs allowed according to the user's role

⇨ Test the distributed command

1. Connect on S1 or S2 as administrator/root
2. Open a system console (PowerShell, shell, …)
3. Change directory to `SAFE`
4. Run `safekit -H "*" level`

   that should return the level for all nodes

11.4.4 Client certificates authentication setup using an external PKI

Client certificates authentication relies on the existence of a set of certificates listed below:

CLCA	The certificate of the Certification Authority CLCA used to issue the client certificates
USER	The client certificates used to assert the user identity and its role in the console ⇨ Personal certificate deployed in the company as the user's digital identity

	Or ⇨ Dedicated certificate generated for the console
ADMIN1 ADMIN2	The client certificates used to assert the administrator identity on S1 and S2 for distributed commands ⇨ Server certificate when it can also be used as client certificate **Or** ⇨ Dedicated certificate generated for distributed commands
PROXY1 PROXY2	The client certificates used to assert the administrator identity on S1 and S2 for the console in proxy mode. They are built from `admin1` and `admin2` certificates.

Apply the following steps to setup user authentication based on client certificates get from your PKI. Previously, HTTPS must have been configured as described in 11.3.2 page 188.

11.4.4.1 Get and install client certificates for the distributed command

11.4.4.1.1 Use server certificates

The server certificates are the one generated during HTTPS configuration described in 11.3.2 page 188.

To verify that these certificates can be used as client certificates, read their contents. Below is the procedure in Windows:

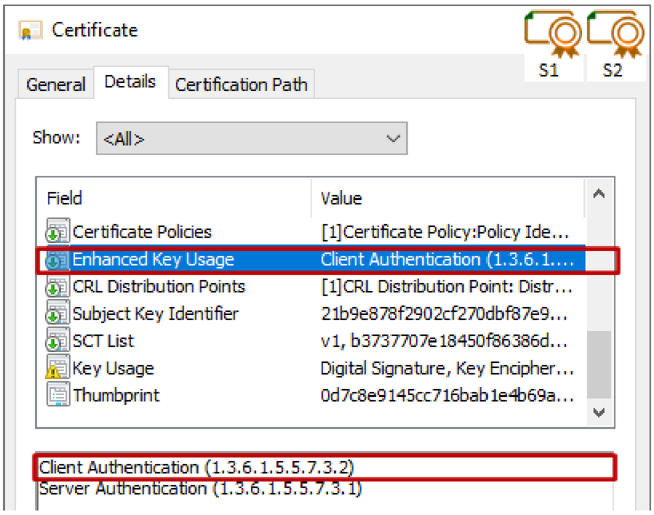
1. Copy the `server.crt` file on a Windows workstation
2. Double click on this file to open it with "Crypto Shell Extensions"
3. Select the "Details" tab
4. Verify the content of the `Enhanced Key Usage` field

Note

If you prefer the command line interface, you can run on each the SafeKit node:

```
SAFE/web/bin/openssl.exe x509 -text -noout -in
SAFE/web/conf/server.crt
```

and look for the value `TLS Web Client Authentication` under the field `X509v3 Extended Key Usage.`



Show: <All>

Field	Value
Certificate Policies	[1]Certificate Policy:Policy Ide…
Enhanced Key Usage	Client Authentication (1.3.6.1.…
CRL Distribution Points	[1]CRL Distribution Point: Distr…
Subject Key Identifier	21b9e878f2902cf270dbf87e9…
SCT List	v1, b3737707e18450f86386d…
Key Usage	Digital Signature, Key Encipher…
Thumbprint	0d7c8e9145cc716bab1e4b69a…

Client Authentication (1.3.6.1.5.5.7.3.2)
Server Authentication (1.3.6.1.5.5.7.3.1)

Important

Note the `CN` value into the `Subject` field, that is used later to configure roles.

Since the `Key Usage` field contains the `Client Authentication` value, this server certificate can be used as client certificate for the distributed command:

ADMIN1	The client certificate to authenticate S1 when running distributed command.
`admin1.crt`	⇨ copy `server1.crt` to `admin1.crt`
`admin1.key`	⇨ copy `server1.key` to `admin1.key`
ADMIN2	The client certificate to authenticate S2 when running distributed command.
`admin2.crt`	⇨ copy `server2.crt` to `admin2.crt`
`admin2.key`	⇨ copy `server2.key` to `admin2.key`

11.4.4.1.2 Use dedicated client certificates

When server certificates cannot be used, you must get new client certificates from you PKI with the expected format described below:

ADMIN1	The client certificate to authenticate S1 when running distributed command.
ADMIN2	The client certificate to authenticate S2 when running distributed command.
`admin1.crt` `admin2.crt`	➩ X509 certificate file in PEM format The `Key Usage` field contains the `Client Authentication` value. The subfield `CN` (`Common Name`) into the `subject` field contains the name of the server. Important: Note the `CN` value, that is used later to configure roles.
`admin1.key` `admin2.key`	➩ The private, *unencrypted* key corresponding to the certificates `admin1.crt`/`admin2.crt`

11.4.4.1.3 Install files in SafeKit

Install the client certificates as follow (where `SAFE=C:\safekit` in Windows if System Drive=C: ; and `SAFE=/opt/safekit` in Linux):

ADMIN1
`admin1.crt`
`admin1.key`
	On S1:
	➩ copy `admin1.crt` to `SAFE/web/conf/admin.crt`
	➩ copy `admin1.key` to `SAFE/web/conf/admin.key`

ADMIN2
`admin2.crt`
`admin2.key`
	On S2:
	➩ copy `admin2.crt` to `SAFE/web/conf/admin.crt`
	➩ copy `admin2.key` to `SAFE/web/conf/admin.key`

PROXY `proxy.crtkey`	On S1 and S2: Build the `SAFE/web/conf/proxy.crtkey` file as follow: ➩ convert `admin.key` in rsa format by running `SAFE/web/bin/openssl rsa -in SAFE/web/conf/admin.key -out SAFE/web/conf/rsa-admin.key`

	⇨ concatenate `admin.crt` and `rsa-admin.key` files into the `proxy.crtkey` file using a text editor or command line

You can check the installed certificates with:

```
cd SAFE/web/bin
checkcert -t client
```

It returns a failure if an error is detected.

11.4.4.2 Get and import client certificates for the web console

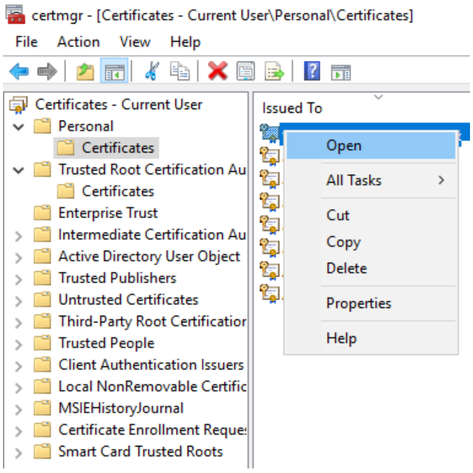
11.4.4.2.1 Use personal certificates

In some companies, each user has a personal certificate as a digital ID, which is stored in the certificate store on the user's workstation.

USER Personal certificate	The personal certificate is used to assert identity of the user in the console

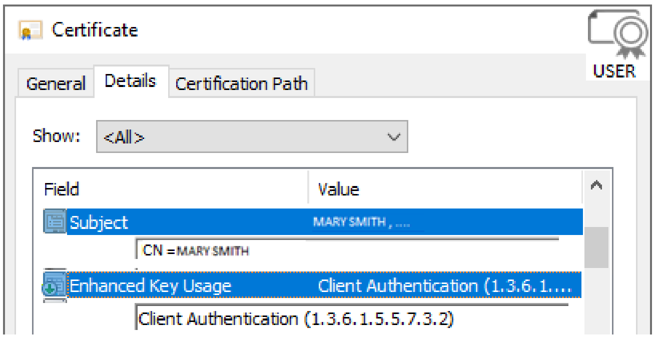
To verify that this certificate can be used as client certificate for the web console, read its contents. Below is the procedure in Windows:

1. Log-in the user’s workstation
2. Open a PowerShell console
3. Run `certmgr`
4. Locate the certificate, typically in “Certificates - Current User\Personal\Certificates”, and right-click
5. Right-click et select “Open” to open the Certificate window


certmgr - [Certificates - Current User\Personal\Certificates]
File
Action
View
Help
Certificates - Current User
Personal
Certificates
Trusted Root Certification Au
Certificates
Enterprise Trust
Intermediate Certification Au
Active Directory User Object
Trusted Publishers
Untrusted Certificates
Third-Party Root Certificatior
Trusted People
Client Authentication Issuers
Local NonRemovable Certific
MSIEHistoryJournal
Certificate Enrollment Reque:
Smart Card Trusted Roots
Issued To
Open
All Tasks
Cut
Copy
Delete
Properties
Help

6. In the Certificate window, select the “Details” tab
7. Verify the content of the `Enhanced Key Usage` field

   For instance, the personal certificate of Mary Smith contains:



Show: <All>

Field | Value
Subject | MARY SMITH , ...
CN = MARY SMITH
Enhanced Key Usage | Client Authentication (1.3.6.1....
Client Authentication (1.3.6.1.5.5.7.3.2)

Important

Note the `CN` value into the `Subject` field, that is used later to configure roles.

Since the `Key Usage` field contains the `Client Authentication` value, this personal certificate can be used as client certificate for the console. In that case, there is no need to import it since it is already present in the certificates store on the user's workstation.

11.4.4.2.2 Use dedicated client certificates

When personal certificates cannot be used, you must get a new client certificate from you PKI with the expected format described below:

USER `user.p12`	The client certificate to authenticate the user of the web console ⇨ X509 certificate file in PKCS#12 format The `Key Usage` field contains the `Client Authentication` value. The subfield `CN` (`Common Name`) into the `subject` field contains the name of the user. Important: Note the `CN` value, that is used later to configure roles.

For each web console user, you must import their certificate into their certificate store. Below is the import procedure in Windows:

1. Log-in the user's workstation
2. Double click on the `user.p12` file for opening the certificate window
3. Click on Install Certificate button

   It opens the Certificate Import Wizard
4. Select Current User and click on the Next button
5. Go on and let the wizard automatically select the certificate store

   It must be the Personal store
6. Then complete the certificate import

11.4.4.3 Get, install, and import the CLCA certificate

11.4.4.3.1 Get certificate file

You must retrieve this certificate from your PKI with the expected format.

CLCA clcacert.crt	The Certification Authority CLCA certificate used to issue the client certificates. ⇨ X509 certificate file in PEM format The chain of certificates for the root and intermediates CLCA If different CLCAs are used to generate the different client certificates, the clcacert.crt file must contain the concatenation of each CLCA certificates.	ADMIN1 ADMIN2 Client certificates for distributed command
		USER Client certificates for web console users

If you have trouble retrieving this file from your PKI, you can build it using the procedure described in 7.18 page 128.

Note

If your PKI uses the same certification authority for issuing server and client certificates, the files cacert.crt and clcacert.crt are identical. The cacert.crt file was installed during the HTTPS configuration procedure (see 11.3.2.2 page 191).

11.4.4.3.2 Install file in SafeKit

Install the certificate as follow (where SAFE=C:\safekit in Windows if System Drive=C: ; and SAFE=/opt/safekit in Linux):

CLCA clcacert.crt	On S1 and S2: ⇨ copy clcacert.crt to SAFE/web/conf/clcacert.crt

You can check the installed certificates:

```
cd SAFE/web/bin
checkcert -t CLCA
```

It returns a failure if an error is detected.

In addition, you must check that the clcacert.crt contains the chain of certificates for the root and intermediates Certification Authorities.

11.4.4.3.3 Import the certificate in the user's certificate store

If the CA certificate has not been imported, the browser issues security alerts when the user connects to the web console with his client certificate. If the import has not already been done, apply the procedure below in Windows:

1. Log-in the user's workstation
2. Click on the `clcacert.crt` file for opening the certificate window
3. Click on Install Certificate button

   It opens the Certificate Import Wizard
4. Select Current User and click on the Next button
5. Go on and install the certificate into the Trusted Root Certification Authorities store
6. Then complete the certificate import

11.4.4.4 Configure roles

The client certificate is used to authenticate the user of the console or of the distributed command. A role must be assigned to it to define the authorized actions.

This must be defined in the file `sslgroup.conf` that can contain the 3 groups Admin, Control, Monitor. Users in these groups will have the corresponding roles.

Note

Each line of the group file must contain the group name followed by a colon, followed by the member users name separated by spaces. See the example above.

Admin Control Monitor sslgroup.conf	⇨ edit the file `sslgroup.conf` ⇨ assign role for each client certificates 1. get the value of the subfield `CN` (`Common Name`) into the `subject` field of the certificate 2. add `CN` with the desired role ✓ for the console certificates, it can be any role: Admin, Control or Monitor ✓ for the distributed command certificates, the role must be Admin
	On S1 and S2 if groups are set: ⇨ copy `sslgroup.conf` to `SAFE/web/conf/sslgroup.conf`

In the following example, `s1.w.com` and `s2.w.com` are the `CN` value for distributed command certificates ; the other names are the `CN` value for console certificates:

Admin
Control
Monitor
sslgroup.conf

```
Admin : "s1.w.com" "s2.w.com" "MARY SMITH" admin
Control: "NAD ROU" "DAVID JOHNS" manager
Monitor : monitor
```

11.4.4.5 Configure and restart the web service

To enable client certificates authentication (where `SAFE=C:\safekit` in Windows if `%SYSTEMDRIVE%`=C: ; and `SAFE=/opt/safekit` in Linux):

httpd.conf	On S1 and S2:
	⇨ edit `SAFE/web/conf/httpd.conf` file
	⇨ comment `usefile` and `useldap`:
	`# Define useldap`
	`…`
	`# Define usefile`
	⇨ verify that only `httpadminport` is defined
	`httpadminport (9010)`
	`#httpcontrolport (9010)`
	`#httpmonitorport (9010)`

	On S1 and S2: ⇨ run `safekit webserver restart`

11.4.4.6 Test the web console and distributed command

The setup is complete; you can now test that it is operational.

⇨ Test the web console

Once certificates are imported on the user workstation, the web console can be used.

1. Start a browser on the user's workstation
2. Connect it to the default URL `http://servername:9010` (where `servername` is the name or Ip address of one of the SafeKit nodes). Since HTTPS is configured, there is an automatic redirection to `https://servername:9453`
3. Depending on the browser and the server IP address, you may sometimes need to select the client certificate to use (the friendly name is displayed)
4. The loaded page contains only the tabs allowed according to the user's role

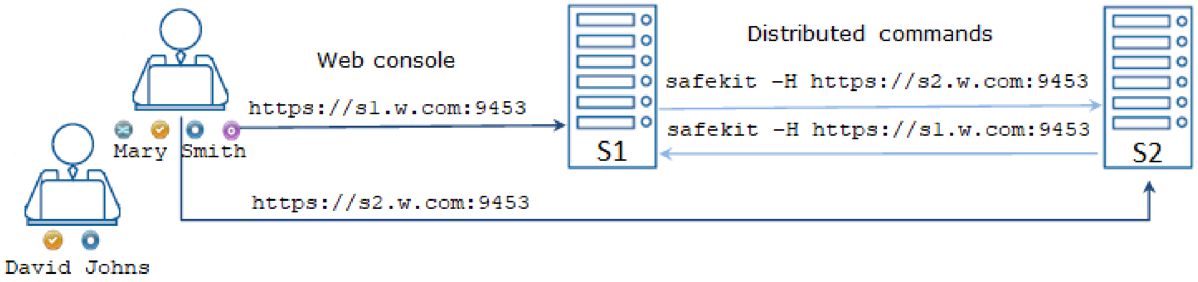
⇨ Test the distributed command

1. Connect on S1 or S2 as administrator/root
2. Open a system console (PowerShell, shell, …)
3. Change directory to `SAFE`
4. Run `safekit -H "*" level`

   that should return the level for all nodes

11.5 Setup example for HTTPS and personal certificate authentication

This section is a summary of the configuration with an external PKI. It shows the configuration of HTTPS and authentication based on users' personal certificates, for the following example:



This simplified setup is only available under certain conditions described below. For all other cases, please refer to 11.3.2 page 188 for the HTTPS setup and to 11.4.4 page 204 for the user authentication setup based on client certificates.

11.5.1 Verify prerequisites

➩ SafeKit cluster configuration

Configure the SafeKit cluster, into `SAFEVAR/cluster/cluster.xml`, as below:

```
<?xml version="1.0" encoding="UTF-8"?>
<cluster>
<lans>
  <lan name="default" console="on" framework="on">
    <node name="s1" addr="s1.w.com"/>
    <node name="s2" addr="s2.w.com"/>
  </lan>
</lans>
</cluster>
```

➩ Server certificates

Request from your PKI, for each SafeKit node, the files for the certificate and the associated key.

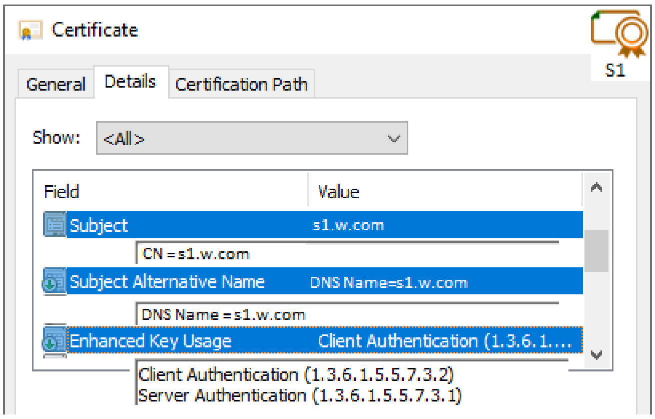
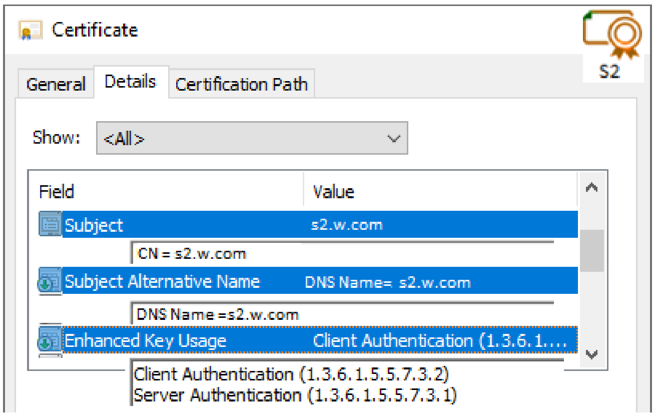
Review the certificate content:

✓ Check that `Enhanced Key Usage` field contains `Client Authentication`

✓ Check that `addr` values in `cluster.xml` is present into the `Subject Alternate Name` field

✓ With SafeKit <= 7.5.2.9, check that server's name is present into the `Subject Alternate Name` field

✓ Note the `CN` value into the `Subject` field, that is used later to fill the `sslgroup.conf` file

Server certificate for S1 `server1.crt` and `server1.key` files	Server certificate for S2 `server2.crt` and `server2.key` files
	

Certificate

S1

General | Details | Certification Path

Show: <All>

Field | Value
Subject | s1.w.com
CN = s1.w.com
Subject Alternative Name | DNS Name=s1.w.com
DNS Name = s1.w.com
Enhanced Key Usage | Client Authentication (1.3.6.1....
Client Authentication (1.3.6.1.5.5.7.3.2)
Server Authentication (1.3.6.1.5.5.7.3.1)

Certificate

S2

General | Details | Certification Path

Show: <All>

Field | Value
Subject | s2.w.com
CN = s2.w.com
Subject Alternative Name | DNS Name= s2.w.com
DNS Name =s2.w.com
Enhanced Key Usage | Client Authentication (1.3.6.1....
Client Authentication (1.3.6.1.5.5.7.3.2)
Server Authentication (1.3.6.1.5.5.7.3.1)

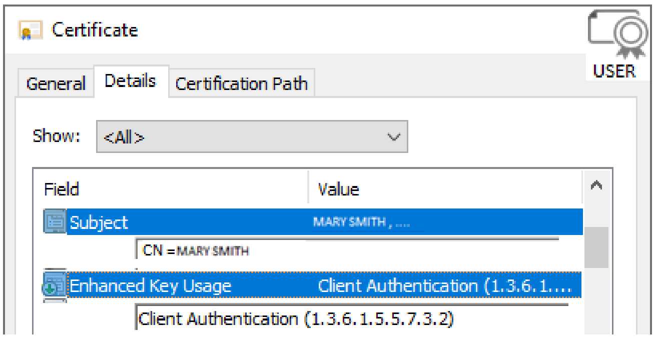
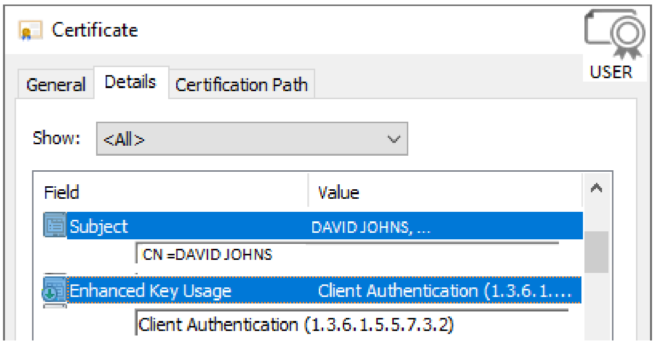
⇨ Personal certificate for the users

It should be present on the user's workstation, inside the certificates store (`certmgr.msc`) under "`Certificates - Current User\Personal\Certificates`"

Review the certificate content:

✓ Check that `Enhanced Key Usage` field contains `Client Authentication`

✓ Note the `CN` value into the `Subject` field, that is used later to fill the `sslgroup.conf` file

Personal certificate for Mary Smith	Personal certificate for David Johns
	

Certificate

USER

General | Details | Certification Path

Show: <All>

Field | Value
Subject | MARY SMITH , ...
CN =MARY SMITH
Enhanced Key Usage | Client Authentication (1.3.6.1....
Client Authentication (1.3.6.1.5.5.7.3.2)

Certificate

USER

General | Details | Certification Path

Show: <All>

Field | Value
Subject | DAVID JOHNS, ...
CN =DAVID JOHNS
Enhanced Key Usage | Client Authentication (1.3.6.1....
Client Authentication (1.3.6.1.5.5.7.3.2)

11.5.2 Setup HTTPS and personal certificate authentication

Apply the following steps to setup HTTPS and client authentication with personal certificates.

11.5.2.1 Get and install certificates in SafeKit

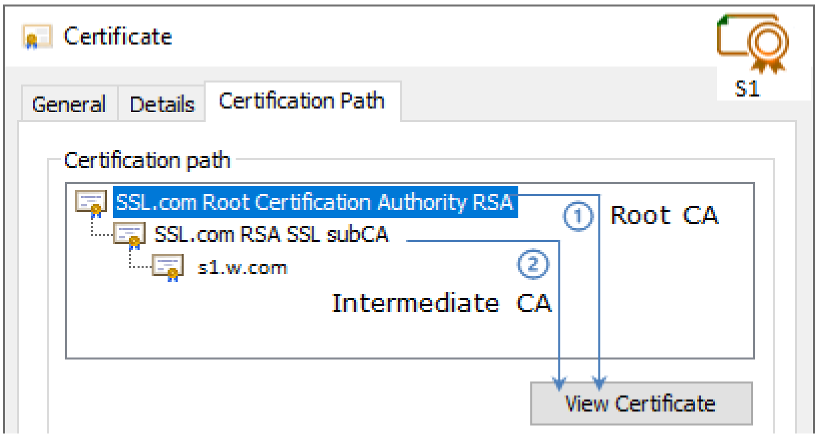
S1
`server1.crt`
`server1.key`
	On S1:
	⇨ Server certificate
	✓ copy `server1.crt` to `SAFE/web/conf/server.crt`
	✓ copy `server1.key` to `SAFE/web/conf/server.key`
	⇨ Client certificate for the distributed command
	✓ copy `server1.crt` to `SAFE/web/conf/admin.crt`
	✓ copy `server1.key` to `SAFE/web/conf/admin.key`

S2
`server2.crt`
`server2.key`
	On S2:
	⇨ Server certificate
	✓ copy `server2.crt` to `SAFE/web/conf/server.crt`
	✓ copy `server2.key` to `SAFE/web/conf/server.key`
	⇨ Client certificate for the distributed command
	✓ copy `server2.crt` to `SAFE/web/conf/admin.crt`
	✓ copy `server2.key` to `SAFE/web/conf/admin.key`

PROXY `proxy.crtkey`	On S1 and S2: ⇨ Client certificate for the proxy mode of the console Build the `SAFE/web/conf/proxy.crtkey` file as follow: ✓ convert `admin.key` in rsa format by running `SAFE/web/bin/openssl rsa -in SAFE/web/conf/admin.key -out SAFE/web/conf/rsa-admin.key` ✓ concatenate `admin.crt` and `rsa-admin.key` files into the `SAFE/web/conf/proxy.crtkey` file using a text editor or command line

CA
cacert.crt

Get the Certification Authority certificate used to issue the server certificates (the certificate chain of the root and intermediates CAs if any). If you do not have it, you can build it as follow:



- ✓ "View Certificate" of the root and intermediates to export them into a file in the "Base-64 encoded X.509 (.CER)" format
- ✓ Concatenate 1, 2 into `cacert.crt`

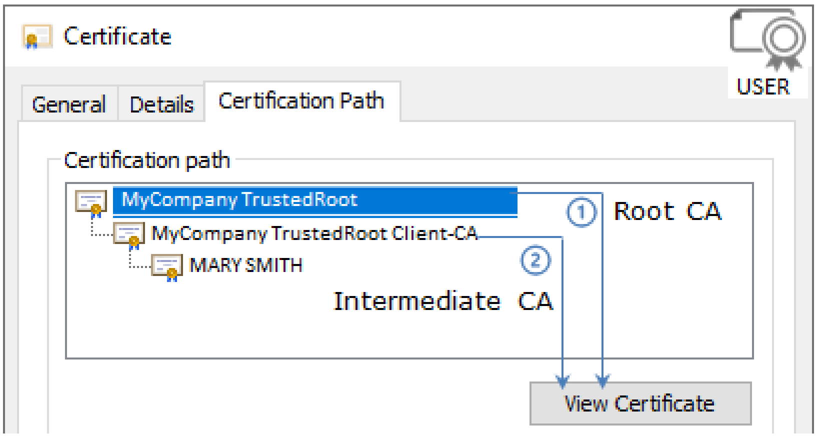
On S1 and S2:

⇨ copy `cacert.crt` to `SAFE/web/conf/cacert.crt`

CLCA

clcacert.crt

Get the Certification Authority certificate used to issue the personal certificates (the certificate chain of the root and intermediates CAs if any). If you do not have it, you can build it as follow:



"View Certificate" of the root and intermediates to export them into a file in the "Base-64 encoded X.509 (.CER)" format. Concatenate 1, 2 into `personalcacert.crt`

On S1 and S2:

⇨ concatenate `cacert.crt` and `personalcacert.crt` into `SAFE/web/conf/clcacert.crt`

11.5.2.2 Configure roles

Admin
Control
Monitor
sslgroup.conf

On S1 and S2:

- edit the file `SAFE/web/conf/sslgroup.conf`
- assign role for each client certificates

```
Admin:"s1.w.com" "s2.w.com" "MARY SMITH"
Control:"DAVID JOHNS"
```

11.5.2.3 Configure and restart the web service

httpd.conf

On S1 and S2:

- edit `SAFE/web/conf/httpd.conf` file
- comment `usefile` and `useldap`:

```
# Define useldap
…
# Define usefile
```

- verify that only `httpadminport` is defined

```
httpadminport (9010)
#httpcontrolport (9010)
#httpmonitorport (9010)
```

httpd.webconsolessl.conf

On S1 and S2:

- copy `SAFE/web/conf/httpd.webconsolessl.conf` to `SAFE/web/conf/ssl/httpd.webconsolessl.conf`

On S1 and S2:

- run `safekit webserver restart`

11.5.2.4 Change the firewall rules

Firewall

On S1 and S2:

- run `SAFEBIN/firewallcfg add`

11.5.3 Test the web console and distributed command

The HTTPS and authentication setup is complete; you can now test that it is operational.

- Test the web console
  1. Start a browser on Mary Smith or David Johns workstation
  2. Connect it to `https://s1.w.com:9453/` or `https://s2.w.com:9453/`

3. Depending on the browser and the server IP address, you may sometimes need to select the client certificate to use (the friendly name is displayed)
4. The loaded page contains only the tabs allowed according to the user's role
   - ✓ Mary Smith, with Admin role, can access to Configuration, Control, Monitoring and Advanced Configuration tabs
   - ✓ David Johns, with Control role, can access to Control and Monitoring tabs

➩ Test the distributed command

1. Connect on S1 or S2 as administrator/root
2. Open a system console (PowerShell, shell, …)
3. Change directory to `SAFE`
4. Run `safekit -H "*" level`

   that should return the level for all nodes

11.6 SafeKit PKI advanced configuration

11.6.1 Configuring with the command line interface

First, choose one SafeKit node to act as the Certificate Authority server. The selected node will be hereafter called the `CA server`. The other cluster nodes are called `non-CA server`. Then go through all the next subsections to activate the HTTPS configuration with the SafeKit PKI.

Important

Verify that the system clock is set to the current date and time on all SafeKit nodes and workstations that will run the HTTPS SafeKit web console. Certificates are timestamped, and a time difference between systems may have an impact on certificate validity.

11.6.1.1 Start the CA web service on the CA server

Apply the same procedure as the one described in 11.3.1.1 page 183, for starting the CA web service (`safecaserv` service).

11.6.1.2 Generate Certificates on the CA server

During this step, the environment for generating certificates is set up: certificate authority, local server and client certificates are created; and server-side certificates are installed in their expected location.

Important

Verify that the system clock is set to the current date and time on the server. Certificates are timestamped, and a time difference between systems may have an impact on certificate validity.

By default, the server certificate includes all the locally defined IP addresses and DNS names. They are listed into the files: `SAFE/web/conf/ipv4.json`, `SAFE/web/conf/ipv6.json` and `SAFE/web/conf/ipnames.json`. These files are built by the command that start the CA web service, called in the previous step.

Important

If the service will be accessed using another DNS name or IP address, edit the corresponding file to insert the new value before executing the `initssl` command. This is required for instance in the clouds using NAT, where the server has a public address mapped on a private address.

On the CA server:

1. Log as administrator/root and open a command shell window
2. Change to the directory `SAFE/web/bin`
3. Run the command:

   ```
   ./initssl ca
   ```

   This command creates a Certificate Authority certificate with the default subject name (that is "SafeKit Local Certificate Authority"). To customize the subject name, run the command with an extra parameter:

   ```
   ./initssl ca "/O=My Company/OU=My Entity/CN=My Company Private
   Certificate Authority"
   ```

   When prompted, enter a password to protect the three default role client certificates and key pairs:

   ```
   Enter the password for the Admin role pkcs12 file
   (../conf/ca/private/user_Admin_administrator.p12) twice:
                             pwd1
                             pwd1


   Enter the password for the Control role pkcs12 file
   (../conf/ca/private/user_Control_manager.p12) twice:
                             pwd2
                             pwd2


   Enter the password for the Monitor role pkcs12 file
   (../conf/ca/private/user_Monitor_operator.p12) twice:
                             pwd3
                             pwd3
   ```

   Note

   The passwords entered at this stage will be needed later to import the client certificates on the client workstation (described in section 11.4.3.4 page 201).

⇨ Optionally, copy the generated client certificates (`.p12` files), that you want to publish, into `../conf/ca/certs`. Also copy the CA certificate `cacert.crt`.

11.6.1.3 Generate certificates on non-CA server

During this step, on non-CA servers, local certificate requests are created, signed certificates are retrieved from the CA server, and finally certificates are installed at their expected locations.

Apply the following procedure sequentially on each non-CA servers:

1. Log on as administrator/root and open a command shell window
2. Change to the directory `SAFE/web/bin`
3. List server DNS names and IP addresses

   By default, the server certificate includes all the locally defined IP addresses and DNS names. They are listed into the files: `SAFE/web/conf/ipv4.json`, `SAFE/web/conf/ipv6.json` and `SAFE/web/conf/ipnames.json`. For building these files, run the command:

   ⇨ In Linux

   ```
   ./getipandnames
   ```

   This command relies on the `host` command delivered with the bind-utils package. Install it if necessary or manually fill the DNS names into the file `SAFE/web/conf/ipnames.json`.

   ⇨ In Windows

   ```
   ./getipandnames.ps1
   ```

Important: If the service will be accessed using another DNS name or IP address, edit the corresponding file to insert the new value before executing the `initssl` command. This is required for instance in the clouds using NAT, where the server has a public address mapped on a private address.

4. Run the command:

   ```
   ./initssl req https://CAserverIP:9001 CA_admin
   ```

   where `CAserverIP` is the DNS name or IP address of the CA server. Then enter, each time it is required, the password you specified when you started the CA web service on the CA server (for instance, `PasW0rD`)

   Or

   ```
   ./initssl req https://CAserverIP:9001 CA_admin:PasW0rD
   ```

   Note: If you get the error "Certificate is not yet valid", it means the system clock of the server is not synchronized with the system clock of the CA server. You should synchronize your server clocks and re-run the `initssl` command if the time difference is not acceptable.

11.6.1.4 Reconfigure the web service for HTTPS on CA server and non-CA server

Once certificates are generated on the CA server and each non-CA servers, the SafeKit web service (`safewebserver` service) can be configured for HTTPS. Apply the procedure described in 10.6.4 page 170, on **all** the servers.

11.6.1.5 Configure the firewall on CA server and non-CA server

When the SafeKit web service runs in HTTPS mode, it is safe to allow network communication with this server and configure the firewall. For this, apply the instructions described in 10.3 page 156.

11.6.1.6 Use the HTTPS SafeKit Web console

1. Download from the CA server the client (`.p12` files) and the CA certificates (`cacert.crt` file) located into `SAFE/web/conf/ca/certs`

2. Import the set of certificates as described in 11.4.3.4 page 201 and 11.4.3.5 page 202

11.6.1.7 Stop the CA web service on CA server

Once all SafeKit nodes and clients have been configured, it is recommended to bring the CA web service (`safecaserv` service) offline on the CA server, to limit the risk of accidental or malicious access to the configuration wizard.

For stopping the SafeKit CA web service with the command line:

1. Log as administrator/root and open a command shell window
2. Change to the directory `SAFE/web/bin`
3. Run the command `./stopcaserv`

Note

On Windows, this command also removes the service entry to prevent any accidental start of the service afterwards. On Linux, the 9001 port is automatically closed on local firewall.

When all foreseeable certificate generation and installation is done, it is a good practice to make sure files unnecessary at production time are not accessible. This step is not mandatory.

The files that constitute the CA, i.e., the `SAFE/web/conf/ca` file tree (especially the private keys stored under `SAFE/web/conf/ca/private/*.key`s) should be stored for future use on a removable storage media and removed from the server. Store the removable media in a secure place (i.e., a vault). This also applies to the files located under the `SAFE/web/conf/ca` directory of non-CA servers. The CA files should be restored into the same location before using the CA again (for example, if adding a new SafeKit cluster node).

11.6.2 Renewing certificates

Every certificate has an expiration date. The default expiration date of the CA certificate is set to 10 years after the CA installation date. The default expiration date of the server and client certificates is set to 5 years after the certificate request date.

Expired client certificates cannot be used to log on to the console, and expired server certificates will trigger warnings when the browser connects to the server. Expired CA certificates cannot be used to validate issued certificates.

It is possible to renew certificates using the original certificate requests and the private keys stored under the `SAFE/web/conf/ca` directory tree. You may also create a new certificate request using the existing private key. The procedure to do so is beyond the scope of this document, see openssl (or your certificate authority) documentation.

Creating a new set of certificates (and private keys) will have the side effect of renewing all certificates. To create a new set of certificates:

1. Erase the `web/conf/ca` directory on all SafeKit servers related to the CA, including the CA SafeKit server itself
2. Suppress existing certificates from the client machines certificate stores
3. Apply the full procedures described in 11.3.1 page 183 and 11.4.3 page 197

11.6.3 Revoking certificates

It is possible to modify the SafeKit web service configuration to use a CRL containing the revoked certificates list. Setting up such a configuration is beyond the scope of this document. Refer to the Apache and openssl documentation.

Creating a new set of certificates and replacing the old set with the new one will have the side effect of effectively revoking the previous certificate set, since the CA certificate is different.

11.6.4 Commands for certificate generation

Commands are located, and must be executed from, the `SAFE/web/bin` directory.

`initssl ca [<subject>]`	**Parameters** `<Subject>`: the CA certificate subject, that identify in human readable form the owner of the CA. **Example** `initssl ca "/O=My Company/OU=My Unit/CN=My Company Private Certificate Authority"` **Description** All paths below are relative to `SAFE/web` directory. This command creates a conf/ca file tree needed for the openssl certificate authority related commands. Generated certificates will be stored in conf/ca/certs. Generated private keys will be stored in conf/ca/private. Note: Note that the best practice is to protect private keys with a password, but it needs more complex configuration on the server and is beyond the scope of this document. See the Apache and OpenSSL documentation for more information. Creates a CA certificate `conf/ca/certs/cacert.crt` and its associated key `conf/ca/private/cacert.key` Creates server default certificate `conf/ca/certs/server_localca.crt` and its corresponding key `conf/ca/private/server_localca.key` Creates the client default certificate for distributed commands `conf/ca/certs/user_Admin_system.crt` and its corresponding key `conf/ca/private/user_Admin_system.key` Creates 3 default certificates corresponding to the 3 predefined roles (Admin, Control, and Monitor) and exports it to pkcs12 file. During this process, the script asks for a password to protect the pkcs12 file, twice for each certificate. The resulting files are: ✓ `conf/ca/private/user_Admin_administrator.p12` ✓ `conf/ca/private/user_Control_manager.p12` ✓ `conf/ca/private/user_Monitor_operator.p12` Client certificates are used as an authentication method on an HTTPS server. They are transmitted to the web service by the browser and verified on the server as part of the HTTPS connection handshake. A

	certificate corresponding to the desired role must be installed in the browser certificate store before the SafeKit web console can be used. Installs the CA certificate, server certificate, and system client certificates in the conf directory

`initssl req <url> <user>[:<password>]]`	**Parameters** `<url>`: Url of the CA service. (https://192.168.0.1:9001) `<user>`,`<password>`: user and password used to authenticate against the CA web service. `<user>` preconfigured value is CA_admin. `<password>` is the one entered by the administrator at the start of CA web service. If this optional field is not present, the password will be asked interactively several times, when needed. **Example** `initssl req https://192.168.0.1:9001 CA_admin:PasW0rD` **Description** All paths below are relative to `SAFE/web` directory. `<hostname>` is the local server's hostname. Creates a certificate request for a server certificate that includes all the locally defined IP addresses and DNS names. The certificate request is stored in `conf/ca/private/server_<hostname>.csr`. The corresponding key is stored in `conf/ca/private/server_<hostname>.key`. Creates a certificate request for a client certificate with the Admin role (to be used by the distributed commands). The certificate request is stored in `conf/ca/private/user_Admin_<hostname>.csr`. The corresponding key is stored in `conf/ca/private/user_Admin_<hostname>.key`. Retrieves the CA certificate from the CA server Retrieves signed certificates corresponding to the certificate requests above, from the CA server (using provided login) Installs certificates and keys Checks certificates are OK

initssl req	**Parameters** None **Description** All paths below are relative to `SAFE/web` directory. In this form, the command stops after having generated the certificate requests corresponding to: The local server, in the `conf/ca/private/server_<hostname>.csr` An Admin role client certificate, in `conf/ca/private/user_Admin_<hostname>.csr` Those certificate requests are stored in a base64 encoded file ready to be submitted to an external certificate authority such as Microsoft Active Directory Certificate Services (refer to the Microsoft documentation on how to submit a base64 encoded certificate request file).

makeusercert `<name>` `<role>`	**Parameters**
	`<name>` is the subject's CN name of the certificate, usually the subject's username.
	`<role>` is subject's role as a console user. The valid value is `Admin` or `Control` or `Monitor`.
	Examples
	`makeusercert administrator Admin`
	`makeusercert manager Control`
	`makeusercert operator Monitor`
	Description
	All paths below are relative to `SAFE/web` directory.
	Creates a client certificate request (and certificate + pkcs12 file containing certificate and key if started on the CA SafeKit server) for the `<name>` and `<role>`.
	When the pkcs12 file is generated, the command asks twice for a password to protect the file. The generated unencrypted private key is stored into `conf/ca/private/user_<role>_<name>.key` file. If applicable, the generated certificate and pkcs12 files are stored into `conf/ca/certs/user_<role>_<name>.crt` and
	`conf/ca/private/user_<role>_<name>.p12` files respectively.

11.6.5 CA web service

The SafeKit CA web service configuration is stored in `SAFE/web/conf/httpd.caserv.conf` file.

This service implements limited PKI functionalities as well as a configuration wizard:

- The configuration wizard is accessible at the https://CAserverIP:9001/ URL.
- Advanced configuration forms related to external PKI use cases are also available.

  https://CAserverIP:9001/advanced.html is a form allowing uploading externally generated certificates and keys to the local server.

  https://CAserverIP:9001/getcsr.html is a form allowing to retrieve locally generated certificates, certificate requests, and p12 files and to request the signature by the local Certification Authority of an externally generated user certificate request (.csr file).

- CA certificates are accessible at the https://CAserverIP>:9001/certs/<certificate name>.crt URL.

  For example, the CA certificate is accessible at https://CAserverrIP>:9001/certs/cacert.crt.

  Certificate signature requests are processed by posting a form at the URL: https://<CA server IP>:9001/caserv.

  The form takes the following parameters:

  action = signrequest

  name = <certificate name>

  servercsr = <file content of the server certificate request>

  Or

  usercsr = <file content of the client certificate request>

12. Cluster.xml for a SafeKit cluster configuration

SafeKit uses the configuration file `cluster.xml`. This file defines all the servers that make up the SafeKit cluster as well as the IP address (or name) of these servers on the networks used to communicate with the cluster nodes. This file also allows specifying the use of networks:

- ✓ a framework network (`framework="on"`) is a network used for internal communications within the SafeKit framework.

  These are global cluster and module internal communications; these communications are encrypted. This network is also used for executing distributed commands. You must define at least one framework network that includes all nodes in the cluster. It is recommended to define several framework networks to tolerate at least one network failure.

- ✓ a console network (`console = "on`") is a network on which the SafeKit web console can connect for cluster and module configuration and administration.

  This type of network must include all the nodes that make up the SafeKit cluster. You can define multiple console networks according to administrative requirements and network topology.

By default, a network is for the console and the framework communications.

12.1 Cluster.xml file

Each network (`lan`) has a logical name that will be used in the configuration of the modules to name the monitoring networks (this network must be configured with `framework = "on"`):

- ➩ into the heartbeat section for a mirror module (for details, see 13.3 page 239)
- ➩ into the lan section for a farm module (for details, see 13.4 page 241)

The node name is the one that is used by the SafeKit administration service (`safeadmin`) for uniquely identifying a SafeKit node. You must always use the same name for designing a given server on different networks. This name is also used by the SafeKit web console when displaying the server name.

12.1.1 Cluster.xml example

In the example below, both networks can be used for the console and the framework communications (by default, `console = "on" framework = "on"`).

```
<cluster>
   <lans>
      <lan name="default">
         <node name="node1" addr="192.168.1.67"/>
         <node name="node2" addr="192.168.1.68"/>
         <node name="node3" addr="192.168.1.69"/>
         <node name="node4" addr="192.168.1.70"/>
      </lan>
      <lan name="repli">
         <node name="node1" addr="10.0.0.1"/>
         <node name="node2" addr="10.0.0.2"/>
         <node name="node3" addr="10.0.0.3"/>
         <node name="node4" addr="10.0.0.4"/>
      </lan>
   </lans>
</cluster>
```

In the example below, the private network cannot be used by the console since it does not include all the nodes in the cluster. This is for example a dedicated replication link for a mirror module.

```
<cluster>
   <lans>
      <lan name="default">
         <node name="node1" addr="192.168.1.67"/>
         <node name="node2" addr="192.168.1.68"/>
         <node name="node3" addr="192.168.1.69"/>
         <node name="node4" addr="192.168.1.70"/>
      </lan>
      <lan name="repli" console="off">
         <node name="node1" addr="10.0.0.1"/>
         <node name="node2" addr="10.0.0.2"/>
      </lan>
   </lans>
</cluster>
```

In the example below, the public network is used only for administration via the console. This is for example a public network that cannot be used for framework communications.

```
<cluster>
   <lans>
      <lan name="default">
         <node name="node1" addr="192.168.1.67"/>
         <node name="node2" addr="192.168.1.68"/>
      </lan>
      <lan name="public" framework=”off”>
         <node name="node1" addr="node1.mydomain.com"/>
         <node name="node2" addr="node2.mydomain.com"/>
      </lan>
   </lans>
</cluster>
```

In the example below, a unique network is used, but in a Network address translation (NAT) configuration. For each node two addresses must be defined: the local one (defined on local interface) and the external one (as seen by other servers)

```
<cluster>
   <lans>
      <lan name="default">
         <node name="node1" addr="server1.dns.name" laddr="10.0.0.1"/>
         <node name="node2" addr="server2.dns.name" laddr="10.0.0.2"/>
      </lan>
   </lans>
</cluster>
```

Notes:

- ✓ All nodes must be able to communicate to the others via the NATted addresses.
- ✓ If a NATted lan is used as console lan, the Web console must be able to communicate to the nodes via the external's addresses.
- ✓ Configuration with NATted addresses could not be done via the Web console, only with the CLI (described in section 12.2.2 page 232).

12.1.2 Cluster.xml syntax

```
<cluster>
  <lans [port="4800"]>
    <lan name="lan_name" [console="on|off"] [framework="on|off"]
[command="on|off"] >
      <node name="node_name" addr="IP1_address"|"IP1_name"
[ laddr="local_IP1_address" ]/>
      <node name="node_name" addr="IP2_address"|"IP2_name"
[ laddr="local_IP2_address" ] />
      …
    </lan>
    …
  </lans>
</cluster>
```

12.1.3 <lans>, <lan>, <node> attributes

`<lans`	Begin the definition of the SafeKit cluster nodes and network topology.
`[port="xxxx"]`	Defines the UDP port with which the membership protocol is exchanged. Default: `4800`
`[pulse="xxxx"]`	Defines the period of the membership protocol messages emission. Longer pulse makes the membership protocol use less bandwidth but react more slowly.
`[mlost_count="xx"]`	Defines the number of periods elapsed without message before electing a new leader.
`[slost_count="xx"]`	Defines the number of periods elapsed without messages before declaring a follower node offline.

`<lan`	Definition of a LAN (i.e., IPv4 broadcast domain, IPv6 link) on which the membership protocol will be transmitted. At least one LAN must be defined. Define one such tag per used LAN.
`name="lan name"`	Single logical name for the lan. This name is used into module configuration to name networks used by the module.
`framework="on"|"off"`	Set `framework="off"` to not use this network for SafeKit framework communications. In this case, this network cannot be used in the configuration of a module. By default, `framework="on"`. You can define multiple `<lan>` sections with `framework="on"` or `framework="off"`. You must define at least one `<lan>` section with `framework="on"`, which includes all nodes in the cluster Default: `on`
`console="on"|"off"`	Set `console="off"` to not use this network for connecting the SafeKit web console. By default, `console ="on"`. When `console="on"`, the `<lan>` section must include all the nodes in the cluster. You can define multiple `<lan>` sections with `console="on"` or `console="off"`. If you want to use the web console, you must define at least one `<lan>` section with `console="on"`. Default: `on`
`command="on"|"off"`	Set `command="on"` to use this network for running distributed commands on the cluster. In this case, this `<lan>` section must include all nodes in the cluster and have `framework="on"`. You can set only one `<lan>` section with `command="on"`. When this attribute is not set, it is the first <`lan>` section with `framework ="on"` that is used for running distributed commands on the cluster. Default: `off`
`<node`	Definition of one node in the SafeKit cluster. Define as many <node> tags as there are nodes in the cluster (at least 2).

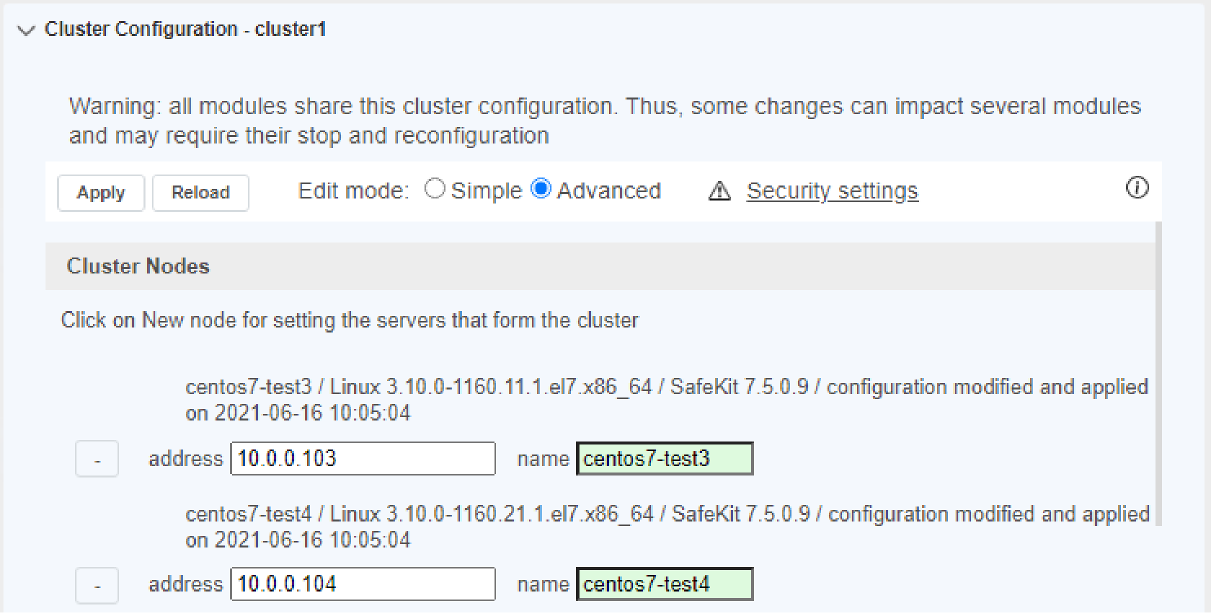
`name="node name"`	Single logical name to the SafeKit server. You must always use the same name for designing a given server on different lans.
`addr= "IP_address"| "IP_name"`	IPv4 or IPv6 address, or name of the node as it is known by other nodes on this LAN (IP address recommended to be independent from a DNS server). On NAT configuration, it must be the external address. When defining an IPv6 address, use literal format: the address is enclosed in square brackets (e.g. `[2001::7334]`)
`laddr= "local_IP_address"`	Local IP address on this LAN. To be used only on NAT configurations, where local address is different from external one. IPv4 address or literal IPv6 address.

12.2 SafeKit cluster Configuration

12.2.1 Configuration with the SafeKit web console

The SafeKit web console provides a graphical user interface for editing the `cluster.xml` file and applying the configuration on all the cluster nodes. See section 3.2 page 37 for a full description.

- Click on › Cluster Configuration to open the panel. It displays the list of cluster nodes
- Edit the configuration. In Simple edit mode, you can edit only the web console connection network. Switch to Advanced edit mode, for editing all the networks.
- Click on the Apply button for saving and applying your configuration on all nodes and generating new key for encrypting communications
- In Simple Edit mode, this button is enabled only if you have done changes. If you want to generate new key or apply the configuration again, switch to the Advanced Edit mode, then click on the Apply button.



Cluster Configuration - cluster1

Warning: all modules share this cluster configuration. Thus, some changes can impact several modules and may require their stop and reconfiguration

Edit mode: ○ Simple ● Advanced ⚠ Security settings

Cluster Nodes

Click on New node for setting the servers that form the cluster

centos7-test3 / Linux 3.10.0-1160.11.1.el7.x86_64 / SafeKit 7.5.0.9 / configuration modified and applied on 2021-06-16 10:05:04

address 10.0.0.103 name centos7-test3

centos7-test4 / Linux 3.10.0-1160.21.1.el7.x86_64 / SafeKit 7.5.0.9 / configuration modified and applied on 2021-06-16 10:05:04

address 10.0.0.104 name centos7-test4

12.2.2 Configuration with command line interface

The commands line equivalent for configuring the SafeKit cluster with a new cryptographic key are:

1. `safekit cluster config [<filepath>]`

   where filepath is the path for the new `cluster.xml`

   when filepath is not set, the current configuration is kept and only encryption key is generated

2. `safekit –H "*" –G`

   it applies the local configuration on all SafeKit nodes defined into cluster.xml

The commands line for re-configuring without cryptographic key are:

1. `safekit cluster delkey`
2. `safekit –H "*" –G`

The commands for re-generating the cryptographic key are:

1. `safekit cluster genkey`
2. `safekit –H "*" –G`

For the full description of commands, refer to 9.3 page 144.

12.2.3 Configuration changes

When changing the SafeKit cluster configuration, the new configuration must be applied on all the servers of the cluster. When the configuration is applied only on a subset of the nodes present into the cluster configuration, only this subset will be able to communicate with each other. This is also the case when the cryptographic key is not identical on all

nodes. This can have the effect of disrupting the operation of the modules installed on servers.
For a correct behavior, you must re-apply the configuration on all the nodes that belong to the cluster as described above.

Note

You can check the configuration by running the command `safekit cluster confinfo` on each node (see section 9.3 page 144). When the configuration is operational, this command must return on all nodes, the same list of nodes and the same value for the configuration signature.

Changing the cluster configuration could have important impact on module configurations since the lan names set into the SafeKit configuration are used into the module’s configuration. Any change in the cluster configuration, will trigger modules updates: each module will reload its configuration to adapt the changes. Such changes could lead to module stop in case of incompatibility (for example if a lan used by a module is removed from the cluster configuration). So, great care must be taken when modifying cluster configuration when modules are running.

13. Userconfig.xml for a module configuration

Important

`userconfig.xml` modifications must be applied to all the servers of the cluster onto which the module is deployed. Apply the new configuration with:

- ✓ web console/ Advanced Configuration /Installed modules/ module/ Apply the configuration
- ✓ or web console/ Configuration/ ▾ on the module/ Edit the configuration
- ✓ or `safekit config -m AM` command (replace AM by the module name)

It is possible to apply a new configuration while the module is running, but only in `ALONE` or `WAIT` (red) states. This feature is called *dynamic configuration*. Only a restricted subset of parameters could be changed. If the new configuration cannot be deployed, an error message is displayed. The attributes that can be dynamically modified are reported hereafter.

Note

Example of `userconfig.xml`

```
<safe>
  <!-- Insert below <macro> <service> tags -->
</safe>
```

13.1 Macro definition (<macro> tag)

13.1.1 <macro> example

```
<macro name="ADDR1" value="aa.bb.com"/>
```

An example of macros usage is given in 15.4 page 303.

13.1.2 <macro> syntax

```
<macro
     name="identifier"
     value="value"
/>
```

13.1.3 <macro> attributes

<macro	
name="identifier"	A character string that identifies the macro.
value="value"	The value that will replace each occurrence of %identifier% in the rest of userconfig.xml.
/>	

Note

The syntax %identifier% can also be used in userconfig.xml to represent the value of an environment variable named identifier. In case of conflict, it is the macro value that is expanded.

13.2 Farm or mirror module (<service> tag)

13.2.1 <service> example

Example for a mirror module:

```
<service mode="mirror" defaultprim="alone" maxloop="3" loop_interval="24"
failover="on">
  <!-- Insert below <hearbeat> <rfs> <vip> <user> <vhost> <errd> <check>
<failover> tags -->
</service>
```

Example for a farm module:

```
<service mode="farm" maxloop="3" loop_interval="24">
  <!-- Insert below <farm> <vip> <user> <vhost> <errd> <check> <failover> tags --
>
</service>
```

See examples of <service> definition for a mirror module in 15.1 page 300 and, for a farm module, in 15.2 page 301.

13.2.2 <service> syntax

```
<service mode="mirror"|"farm"|"light"
  [boot="off"|"on"|"auto"|"ignore"]
  [boot_delay="0"]
  [failover="on"|"off"]
  [defaultprim="alone"|"server_name"|"lastprim"]
  [maxloop="3"] [loop_interval="24"]
  [automatic_reboot="off"|"on"]>
</service>
```

Note

Only `boot`, `maxloop`, `loop_interval` and `automatic_reboot` attributes can be changed with a dynamic configuration.

13.2.3 <service> attributes

`<service`	Top level section of `userconfig.xml`

`mode=`
`"mirror"|`
`"farm"|`
`"light"`
	`The mirror` keyword sets the module behavior to mirror architecture mode. The synchronization protocol between the 2 servers is defined in section 13.3 page 239.
	See mirror.safe application module for an example.
	`The farm` keyword sets the module behavior to farm architecture mode. The definition of the synchronization protocol between servers is described in section 13.4 page 241.
	See farm.safe application module for an example.
	`The light` keyword sets the module behavior to the minimum needed for one server with software error detection and local restart only

`[boot=`
`"on"|`
`"off"|`
`"auto"|`
`"ignore"]`
	If set to `on`, the module is automatically started at boot time.
	If set to `off`, the module is not started at boot time.
	If set to `auto`, the module is automatically started at boot time, if it was started before the reboot.
	Before SafeKit 7.5, the configuration to start the module at boot was done with the command `safekit boot -m AM on | off` (which had to be executed on each node). If you prefer to continue using this command, remove the `boot` attribute or set it to `ignore` (the default). The module will not be started at boot time unless the `safekit boot -m AM on` command is executed.
	The state of the boot configuration is visible in the `usersetting.boot` resource. The status of resources is visible in web console/ Control/Select the node/Resources tab/; with the command `safekit state -m AM -v`
	Default value: `ignore`

`[boot_delay="0"]`	The delay, in seconds, before starting the module at boot. Default value: `0` (no delay)

<table>
<tr><td>[failover=
"on"|
"off"]</td><td>For mirror module only.
If set to on, an automatic failover on the secondary server is triggered if the primary fails or stops.
If set to off, when the primary server fails or stops, the secondary server waits (no automatic failover is triggered). Only the prim command can start the secondary server as primary. See description in 5.7 page 101
Default value: on</td></tr>
<tr><td>[defaultprim=
"alone"|
"server_name"|
"lastprim"]</td><td>For mirror module only.
defaultprim specifies which server among two servers is the default primary server for an application module.
This option is useful when a module is ALONE on a server and the module is started on the other server.
With defaultprim="alone", the ALONE module becomes PRIM while the module on the other server becomes SECOND. Value recommended avoiding swap of application after reintegration.
With defaultprim="server_name", when the module is running on two servers, the primary server among the two servers is the one set in defaultprim. This value can be useful for active/active or N-1 architectures see section 1.5.1 page 20 or section 1.5.2 page 21.
With defaultprim="lastprim", the restarted module becomes PRIM if it was PRIM before its last stop.
Default value: alone</td></tr>
<tr><td>[maxloop="3"]</td><td>Number of successive error detections before stop.
This attribute defines the maximum number of "restart" or "stopstart" sequences that can be automatically triggered by failure detectors before the module locally stops.
The counter is reset to its initial value at the expiration of the loop_interval timeout and upon safekit start, restart, swap, stopstart… administrative commands execution.
Note that a safekit command sent by a detector passes the -i identity parameter and decrements the counter, whereas administrator issued commands do not.
For more information, see 13.18.4 page 289.
Note This attribute’s value can be changed with a dynamic configuration.
Since SafeKit 7.5, the maxloop is represented by the resource heart.stopstartloop. Its current value corresponds to the date on which the counter was initialized (in the form of a Unix Epoch timestamp); and its assignment date corresponds either to its</td></tr>
</table>

	initialization or to a `stopstart`, `restart`. View the resource history to see each increment of the loop counter. Default value: `3`
`[loop_interval ="24"]`	Time interval during which `maxloop` applies. If set to 0, the `maxloop` counter becomes inactive. Default value: 24 hours. Note This attribute's value can be changed with a dynamic configuration.
`[automatic_reboot ="off"| "on"]`	If set to `on`, "stopstart" triggers a reboot instead of stopping and restarting the module. Default value: `off` Note This attribute's value can be changed with a dynamic configuration.

13.3 Heartbeats (<heart>, <heartbeat > tags)

Heartbeats must be used only for mirror architecture. For farm architecture, see section 13.4 page 241.

The basic mechanism for synchronizing two servers and detecting server failures is the heartbeat, which is a monitoring data flow on a network shared by a pair of servers. Normally, there are as many heartbeats as there are networks shared by the two servers. In normal operation, the two servers exchange their states (PRIM, SECOND, the resource states) through the heartbeat mechanism and synchronizes their application start and stop procedures.

If all heartbeats are lost, it is interpreted as if the other server was down, and the local server switches to the ALONE state. Although not mandatory, it is better to have two heartbeat channels on two different networks for synchronizing the two servers to avoid the split-brain case.

13.3.1 <heart> example

```
<heart>
   <heartbeat name=”default” ident="Hb1" />
   <heartbeat name=”net2” ident="Hb2" />
</heart>
```

13.3.2 <heart> syntax

```
<heart
  [port="xxxx"] [pulse="700"] [timeout="30000"]
  [permanent_arp="on"]
```

```
>
 <heartbeat
  [port="xxxx"] [pulse="700"] [timeout="30000"] name=”network” [ident="name"]
 >
 [<!-- syntax for SafeKit < 7.2 -->
  <server addr="IP1_address"|"IP1_name" />
  <server addr="IP2_address"|"IP2_name" />
 ]
</hearbeat>
 …
</heart>
```

Note

The `<heart>` tag and full subtree can be changed with a dynamic configuration.

13.3.3 <heart>, <heartbeat > attributes

<heart	
[port="xxxx"]	UDP port on which all the heartbeats are exchanged. Default: depends on the id of the application module. Returned by the `safekit module getports` command.
[pulse="700"]	The delay, in milliseconds, between two heartbeat packets. Default value: 700 ms
[timeout="30000"]	Timeout value for heartbeat loss detection. Default value: 30 000 ms
<heartbeat	Definition of one heartbeat. There are as many `<heartbeat>` tags as there are networks used to probe servers’ mutual connectivity. At least one heartbeat must be defined.
[port="xxxx"]	Redefines the UDP port for the heartbeat. Default value is the same as the one defined in <heart> tag.
[pulse="700"]	Redefines the delay in milliseconds between two heartbeat packets. Default value is the same as the one defined in <heart> tag.
[timeout= "30000"]	Redefines the timeout value for heartbeat loss detection. Default value is the same as the one defined in <heart> tag.
name="network"	Network named used by the heartbeat. `network` must be the name of a network set into the SafeKit cluster configuration (for details, see 12 page 227). This attribute is mandatory in new config syntax (since SafeKit 7.2).

[ident="name"]	Set how the heartbeat will be labelled in the web console and in internal "resources", i.e.: The internal resource heartbeat.name can be used in the failover machine described in 13.18 page 288. If no ident attribute is present the value of the name attribute will be used. Important: ident="flow" is a reserved name associated with a heartbeat declared on a replication flow. If you set a heartbeat with ident="flow", automatically the replication flow will be set on the same network. If you set ident="flow" without <rfs> configuration, the module start blocks in WAIT state.
[permanent_arp= "on"|"off"]	Regularly, heart sets a permanent ARP entry for the ip addresses associated with the heartbeats. On some Linux systems, it may cause heart to freeze. Set this parameter to off in this case and manually set permanent arp for the remote server on boot. On Linux, this can be done by inserting the following line into a script that is executed at boot: arp -s hostname hw_addr Default value: on
[<server addr= "IP1_address />]	Definition of the server address in the heartbeat. The <server> tag is a legacy syntax used in previous SafeKit version (before SafeKit 7.2). It's supported for compatibility reason but must not be used for new modules. Important: In the same userconfig.xml, you must not use the syntax for SafeKit 7.1 and the one for SafeKit 7.2.

13.4 Farm topology (<farm>, <lan> tags)

The basic mechanism to synchronize a farm of servers is a group communication protocol which automatically detects the available members of the farm. Normally, the membership protocol is configured on all networks connecting the N servers.

13.4.1 <farm> example

```
<farm>
   <lan name="default" />
   <lan name="net2" />
</farm>
```

For examples of <farm> configuration, see section 15.5 page 304.

13.4.2 <farm> syntax

```
<farm [port="xxxx"]>
   <lan name=”network” >
     [<!-- syntax for SafeKit < 7.2 -->
      <node name="server1" addr="IP1_address" />
      <node name="server2" addr="IP2_address" />
     ]
   </lan>
   …
</farm>
```

Note

The `<farm>` tag and subtree cann**ot** be changed with a dynamic configuration.

13.4.3 <farm>, <lan> attributes

`<farm`	Begin the definition of a farm topology.
`[port="xxxx"]`	UDP port with which the membership protocol is exchanged. Default: depends on the id of the application module. Returned by the command `safekit module getports`.
`[pulse=”xxxx”]`	The period of the membership protocol messages emission. Longer pulse makes the membership protocol use less bandwidth but reacts more slowly.
`[mlost_count=”xx”]`	Number of periods elapsed without message before electing a new leader.
`[slost_count=”xx”]`	Number of periods elapsed without messages before declaring a follower node offline.
`<lan`	Definition of a LAN (i.e., IPv4 broadcast domain, IPv6 link) on which the membership protocol will be transmitted. At least one LAN must be defined. Define one such tag per used LAN.
`name="network"`	Define the name of network used. `network` must be the name of a network set into the SafeKit cluster configuration (see 12 page 227). This attribute is mandatory in new config syntax (since SafeKit 7.2).
`[<node name=”identity” addr= "IP1_address" />]`	Address and name of the node on this lan. The node tag is a legacy syntax used in previous SafeKit version (before SafeKit 7.2). It’s supported for compatibility reason but must not be used for new modules. Important In the same `userconfig.xml`, you must not use the syntax for SafeKit 7.1 and the one for SafeKit 7.2.

13.5 Virtual IP address (<vip> tag)

Important

If you install and run several application modules on the same server, the virtual IP addresses must be different for each application module.

13.5.1 <vip> example in farm architecture

The following example configures load balancing to port 80 and virtual IP address between nodes in an on-premises cluster:

```
<vip>
    <interface_list>
      <interface check="on" arpreroute="on" arpinterval="60" arpelapse="10">
        <virtual_interface type="vmac_directed">
          <virtual_addr addr="192.168.1.222" where="alias" check="on"/>
        </virtual_interface>
      </interface>
    </interface_list>
    <loadbalancing_list>
       <group name="FarmProto">
         <rule port="80" proto="tcp" filter="on_port"/>
       </group>
    </loadbalancing_list>
  </vip>
```

See also the example in section 15.2 page 301.

13.5.2 <vip> example in mirror architecture

The following example configures the virtual IP address on the primary node of an on-premises cluster:

```
<vip>
    <interface_list>
        <interface check="off" arpreroute="on">
           <real_interface>
               <virtual_addr addr="192.168.1.222" where="one_side_alias"
check="on"/>
          </real_interface>
        </interface>
    </interface_list>
  </vip>
```

See also the example in 15.1 page 300.

13.5.3 Alternative to <vip> for servers in different networks

The configuration of a virtual IP address with a <vip> section in `userconfig.xml` requires servers in the same IP network (network rerouting and load balancing made at level 2).

If servers are in different IP networks, the <vip> section cannot be configured. In this case, an alternative is to configure the virtual IP in a load balancer. The load balancer routes packets to the physical IP addresses of servers by testing an URL status named health check and managed by SafeKit.

So, SafeKit provides a health check for SafeKit modules. For this, configure the health check in the load balancer with:

- HTTP protocol
- port 9010, the SafeKit web service port
- URL /var/modules/AM/ready.txt, where AM is the module name

In a mirror module, the health check:

- returns OK, that means that the instance is healthy, when the module state is ✓ PRIM (green) or ✓ ALONE (green)
- returns NOT FOUND, that means that the instance is unhealthy, in all other states

In a farm module, the health check:

- returns OK, that means that the instance is healthy, when the farm module state is ✓ UP (green)
- returns NOT FOUND, that means that the instance is out of service, in all other states

Another alternative is that you implement a special DNS configuration and a DNS rerouting command inserted in the SafeKit restart scripts.

13.5.4 <vip> syntax

13.5.4.1 Virtual IP loadbalancing in farm architecture

```
<vip [tcpreset="off"|"on"]>
 <interface_list>
  <interface
    [check="off"|"on"]
    [arpreroute="off"|"on"]
    [arpinterval="60"]
    [arpelapse="1200"]
  >

   <virtual_interface
   [type="”vmac_directed”|”vmac_invisible”]
   [addr="xx:xx:xx:xx:xx"]
   >
     <virtual_addr
      addr="virtual_IP_name"|"virtual_IP_address"
      [where="alias"]
      [check="off"|"on"]
      [connections="off"|"on"]
     />
     …
   </virtual_interface>
  </interface>
 </interface_list>
 <loadbalancing_list>
   <group name="group_name"
     <cluster>
        <host name="node_name" power="integer" />
        …
     </cluster>
     <rule
       [virtual_addr="*"|"virtual_IP_name"|"virtual_IP_address"]
       [port="*"|"value"]
        proto="udp"|"tcp"
        filter="on_addr"|"on_port"|"on_ipid"
     />
     …
   </group>
   …
 </loadbalancing_list>
</vip>
```

Note

The <vip> tag and subtree **cannot** be changed with a dynamic configuration.

13.5.4.2 Virtual IP failover in mirror architecture

For on-premises SafeKit cluster:

```
<vip [tcpreset="off"|"on"]>
 <interface_list>
  <interface
    [check="off"|"on"]
    [arpreroute="off"|"on"]
    [arpinterval="60"]
    [arpelapse="1200"]
  >

   <real_interface>
    <virtual_addr
      addr="virtual_IP_name"|"virtual_IP_address"
      where="one_side_alias"
      [check="off"|"on"]
      [connections="off"|"on"]
    />
    …
   </real_interface>
  </interface>
  …
 </interface_list>

</vip>
```

13.5.5 <vip><interface_list>, <interface>, <virtual_interface>, <real_interface>, <virtual_addr> attributes

<vip	

`[tcpreset="off"|"on"]`	Before unconfiguring the virtual IP address, all connections with the virtual IP address as IP source are reset. The reset is disabled when set to `off`. Default value: `on`
`<interface_list>`	
`<interface`	Definition of an interface with virtual IP addresses. Define as many <interface> sections as there are network interfaces to configure.
`[check="off"|"on"]`	Set an interface checker on the interface to stop the service and put it in the WAIT state when the interface is down. The name of the interface checker is `intf.<network_IP_mask>` (`intf.192.168.0.0`). Default value: `on` For more information, see 13.13 page 281.
`[arpreroute="off"|"on"]`	Automatically broadcast gratuitous ARP on virtual IP addresses defined in <real_interface> section. Default value: `off`.
`[arpinterval="60"]`	Time in seconds between two gratuitous ARP. Default value: `60` s
`[arpelapse="1200"]`	Time during which gratuitous ARP are sent. Default value: `1200` s
`[name="interface name"]`	Linux only. You can specify the name of the network interface on which the virtual IP addresses will be set. Ex.: `name="bond0"` Default: no value, SafeKit detects the network interface with virtual IP addresses set on it.

13.5.5.1 <virtual_interface>, <virtual_addr> attributes in farm architecture

Use with farm modules for virtual IP load-balancing:

`<virtual_interface`	Definition of virtual IP addresses configured on an Ethernet interface.

`type=`
`"vmac_directed"|`
`"vmac_invisible"`
	`vmac_directed`: advertise the MAC address of one of the servers as the associated mac address, as with normal traffic. No promiscuous mode needed. For details, see 13.5.7.3 page 250.
	`vmac_invisible`: virtual MAC address never visible in Ethernet headers to allow broadcasting of switch.

	Needs promiscuous mode. For details, see 13.5.7.2 page 250 Note: can be used for a mirror module with a need of transparent rerouting.
[addr="xx:xx:xx:xx:xx"]	Unicast virtual MAC address value. If not set, default is the concatenation of "5A:FE" (Safe) and the first configured virtual IP address in hexadecimal. Ignored in vmac_directed mode.
<virtual_addr	Definition of one Virtual IP address. Set as many <virtual_addr> sections as there are virtual IP addresses on the interface.
addr="virtual_IP_name"| "virtual_IP_address"	Name or address of the virtual IP (prefer an IP address to be independent from the name server). IPv4 or IPv6 address.
where="alias"	Configuration for farm module: the virtual IP address is defined on all servers as an alias IP address. Load balancing rules apply only for this type of virtual IP addresses. Note : when VMAC is used with a mirror module, set here where="one_side_alias"
[check="off"|"on"]	Defines an ip checker on the virtual IP address to stopstart the module when the virtual IP is deleted or in conflict. The name of the ip checker is ip.<addr value> (ip.192.168.1.99). Default value: on For more information, see 13.14 page 282
[connections="off"|"on"]	Enables counting of the number of active connections on the virtual address. This count is stored in the resource named connections.<virtual addr value> (for example: connections.192.168.1.99) which is assigned every 10 seconds. This value is provided as a guideline only. Default value: off
netmask="defaultnetmask"	Linux and IPV4 only. By default, the netmask of the network interface on which the virtual IP address is set. Set the netmask if there are several netmasks on the interface.
</virtual_interface>	

13.5.5.2 <real_interface>, <virtual_addr> attributes in mirror architecture

Use with mirror modules for virtual IP failover:

`<real_interface>`	Definition of virtual IP addresses associated with the real MAC address of the interface.
`<virtual_addr`	Definition of one virtual IP address. Set as many `virtual_addr` sections as there are virtual IP addresses on the interface.

`addr=`
`"virtual_IP_name"|`
`"virtual_IP_address"`
	Name or address of the virtual IP (prefer an IP address to be independent from the name server).
	IPv4 or IPv6 address.

`where="one_side_alias"`	The Virtual IP address will be aliased on the server on which the module becomes `PRIM` or `ALONE`.
`[check="off"|"on"]`	Defines an ip checker on the virtual IP address to stopstart the module when the virtual IP is deleted or in conflict. The name of the ip checker is `ip.<addr value>` (`ip.192.168.1.99`). Default value: `on` For more information, see 13.14 page 282.
`[connections="off"|"on"]`	Enables counting of the number of active connections on the virtual address. This count is stored in the resource named `connections.<virtual addr value>` (for example: `connections.192.168.1.99`) which is assigned every 10 seconds. This value is provided as a guideline only. Default value: off
`netmask="defaultnetmask"`	Linux and IPV4 only. By default, the netmask of the network interface on which the virtual IP address is set. Set the netmask if there are several netmasks on the interface.
`</real_interface>`	

13.5.6 <loadbalancing_list>, <group>, <cluster>, <host> attributes

For load-balancing examples, see 15.5 page 304.

Use with farm module.

`<loadbalancing_list>`	

`<group`	Definition of a load balancing group. Define as many sections as there are groups. An example is given in 15.5.3 page 305.
`name="group_name"`	Name of the load balancing group.
`<cluster`	Definition of the server set on which the load current group balancing will be applied. If no `<cluster>` section is defined, the rules apply to all servers of the farm.
`<host`	Definition of one node in the cluster. Define as many hosts sections as there are nodes configured for the module.
`name = "node_name"`	Define the name of the host. `node_name` must be the name of a node name set into the SafeKit cluster configuration (see 12 page 227).
`power = "value"`	Relative weight to apply to the current node in this load balancing group's cluster. Can be equal to 0, which means no traffic will be dispatched to this node. See section 13.5.7.4 page 250 for more information.
`</cluster>`	
`<rule`	Definition of a load balancing rule for the group. Define as many sections as there are load balancing rules for this group.
`[virtual_addr= "*" | "virtual_IP_address"| "virtual_IP_name"]`	Virtual IP name or address scope of the rule. By default, all virtual IP addresses: `*`
`[port="*"|"value"]`	TCP or UDP port to which the load balancing rule applies. By default, all ports: `*`

`proto="udp" | "tcp" | "arp"`	`proto="udp"`
	Load balancing rule applies to the UDP protocol.
	`proto="tcp"`
	Load balancing rule applies to the TCP protocol.
	`proto="arp"`
	Load balancing rule applies to the IP<->MAC resolution protocol (arp or neighbour discovery)

`filter="on_addr" | "on_port" | "on_ipid"`	`filter="on_addr"`
	Load balancing criteria is the source IP address (client, far end of the connection) (see 15.5.1 page 304).
	`filter="on_port"`
	Load balancing criteria is the source port (client, far end of the connection) (see 15.5.1 page 304).
	`filter="on_ipid"`
	Load balancing is made on the client ip_id at input.

	Useful for UDP. No sense for TCP and for IPv6 addresses (see example in 15.5.2 page 305).

13.5.7 <vip> Load balancing description

13.5.7.1 <vip> prerequisites

See network prerequisites described in 2.3.2 page 30.

13.5.7.2 What is the vmac_invisible type?

When `type=“vmac_invisible”`, a virtual MAC address is mapped on the virtual IP address with a unicast MAC Ethernet address on several network nodes. When a network device tries to resolve the virtual IP address into its corresponding MAC address, the SafeKit servers respond with the virtual MAC address. However, SafeKit servers use its physical MAC address to communicate. To “see” the packets sent to the virtual MAC address the interface is set to promiscuous mode. So, the virtual MAC address is invisible to layer 2 network devices. Ethernet switches therefore forward virtual MAC address directed packets to all the ports in the same vlan as the source, reaching all the servers of the farm. A kernel module running on each farm server is responsible for filtering out the packets that should not be processed by a given farm node, according to the load balancing rules defined.

With the virtual MAC address technology, the failover time is null. There is no network rerouting after a failure: all network equipment keeps their mapping virtual IP address, virtual MAC address.

To test a virtual MAC address in your network, see 4.3.7 page 84

13.5.7.3 What is the vmac_directed type?

When `type=“vmac_directed”`, there is in fact no virtual MAC address. Farm servers reply to virtual IP resolution requests with their own physical MAC address. A kernel module running on each farm server is responsible for filtering and dispatching the packets to their designated target farm node according to the load balancing rules defined. In vmac_directed mode there is a short failover time for clients that have resolved the virtual IP address as the MAC address of the failed server. This is comparable to what happens in “real interface” mode. Clients that have another farm server’s MAC address in their cache are not affected.

To help minimize failover time in ipv4, set the arpreroute attribute to “on” on the corresponding “<interface>” tag, and tune the arpelapse and arpinterval attributes to the desired values. Ipv6 does not need arpreroute, it has a built-in mechanism that takes care of the failover.

13.5.7.4 How does load balancing work?

On all the servers of the farm, the load balancing algorithm filters received packets according to the identity of the sender. The criteria to check is defined by configuration in `userconfig.xml`: client IP address, client port… (i.e.: level 3 load balancing), or requestor address (arp rules, i.e., level 2 load balancing). The criteria are hashed into a value representing the server on which the packet is to be accepted.

When a server fails, the membership protocol reconfigures the filters to re-balance the traffic of the failed server on the available servers.

Each server can have a power (=1, 2…) and then takes more or less traffic. The power is implemented by the number of bits set to 1 in the hash table (a bitmap of 256 bits).

A bitmap example is given in 4.3.5 page 82.

13.6 File replication (<rfs>, <replicated> tags)

For mirror modules only.

In Linux, you must set the same value for uid/gid on the two nodes for replicating file permissions. When replicating a filesystem mount point, you must apply a special procedure described in 13.6.4.2 page 259.

In Windows, it is strongly recommended to enable the USN journal on the drive that contains the replicated directory as described in 13.6.4.3 page 261.

Important

If you install and run several application modules on the same server, the replicated directories must be different for each application module.

13.6.1 <rfs> example

Example in Windows:

```
<rfs async="second">
         <replicated dir="c:\safedir" mode="read_only"/>
</rfs>
```

Example in Linux:

```
<rfs async="second">
         <replicated dir="/safedir" mode="read_only"/>
</rfs>
```

See also the example in 15.4 page 303.

13.6.2 <rfs> syntax

```
<rfs
     [acl="on"|"off"]
     [async="second"|"none"]

     [iotimeout="nb seconds"]
     [roflags="0x10"|"0x10000"]
     [locktimeout="100"]
     [sendtimeout="30"]

     [nbrei="3"]
     [ruzone_blocksize="8388608"]
     [namespacepolicy="0"|"1"|"3"|"4"]
     [reitimeout="150"]
     [reicommit="0"]
     [reidetail="on"|"off"]
     [allocthreshold="0"]
     [nbremconn ="1"]

     [checktime="220000"]
     [checkintv="120"]
     [nfsbox_options="cross"|"nocross"]
     [scripts="off"]
     [reiallowedbw=”20000”]
     [syncdelta=”nb minutes”]
     [syncat=”synchronization scheduling”]
>
  [<flow name=”network” >
    [<!-- syntax for SafeKit < 7.2 -->
     <server addr="IP_address_1" />
     <server addr="IP_address_2" />
    ]
  </flow>]
  <replicated dir="absolute path of a directory"
  [mode="read_only"]
>
  <tocheck path="relative path of a file or subdir" />
  <notreplicated path="relative path of a file or subdir" />
  <notreplicated regexpath="regular expression on relative path of a file or
subdir" />
  …
 </replicated>
</rfs>
```

Note

Only `async`, `nbrei`, `reitimeout` and `reidetail` attributes of `<rfs>` tag can be changed with a dynamic configuration. The `<flow>` tag, describing the replication flow, can also be changed dynamically.

13.6.3 <rfs>, <replicated> attributes

`<rfs`	
`[mountoversuffix = "`*`suffix`*`"]`	Linux only. During the module configuration, the replicated directory "`/a/dir`" is renamed "`/a/dir`*`suffix`*". The directory **`/a/dir`** is created and it is: ➩ a mount point to `/a/dir`*`suffix`* when the module is started ➩ a link to "`/a/dir`*`suffix`*" when the module is stopped By default, *suffix* value is “_For_SafeKit_Replication”. Note If there is a hard failure, then the symbolic link will not be restored. In this case, you must restore the symbolic link manually.

	Important **Restriction** You cannot directly specify a root file system as a replicated directory (because of the directory rename that is not allowed across a file system). The work around is to manipulate the fstab file as described in a KB on https://support.evidian.com. Important When the module is started, NEVER ACCESS files in "/a/dir*suffix*", otherwise the modifications will not be replicated, and the system will become inconsistent. ALWAYS ACCESS replicated files through "/a/dir".
[acl= "on" | "off"]	Setting acl to on activate the replication of ACL on files and directories. Default value: off Importan **Restrictions for Windows** ACL replication will not work if the SYSTEM account does not have the "Full control" access right on all the replicated forest. File ACLs are replicated literally (as SID values), therefore ACL granted to locally defined users and groups will be meaningless on the remote system. File encryption and file compression attributes are not supported.
[async= "second" | "none"]	Setting async mode to second is a way to improve file replication performances: modification operations are cached on the secondary server and the acknowledgements are sent more quickly to the primary server. Setting async mode to none ensures more robustness: modification operations are put on disk of the secondary before sending acknowledgement to the primary. With async="second", in case of double failure at the same time of both PRIM and SECOND servers, if the PRIM server cannot restart, then the SECOND server does not have up-to-date data on its disk. There is data loss if the SECOND server is forced to start as primary with the prim command. Default value: second Note This attribute’s value can be changed with a dynamic configuration.
[packetsize]	Linux only. Maximum size in bytes for NFS replication packets. It must be lower than the maximum size allowed by the NFS server of both servers.

	When it is set into the configuration, it is used as mount options for rsize and wsize. By default, the size is the one of the NFS server.
`[reipacketsize="8388608"]`	Maximum size in bytes of reintegration packets. In Linux, this value must be less or equal to `packetsize`. Default value in Linux: value of `packetsize` if it is set into the configuration and is lower than `8388608`; else `8388608` Default value in Windows: `8388608` bytes
`[ruzone_blocksize="8388608"]`	Size of a zone for the modification bitmap of a file. It must be a multiple of `reipacketsize` attribute. Default value: value of `reipacketsize` if it is set into the configuration; else `8388608`
`[iotimeout]`	Windows only. IO time out in seconds in the Windows file system filter. If an IO cannot be replicated and if the timeout expires in the filter, then the `PRIM` server becomes `ALONE`. If not set, the default value is dynamically calculated.
`[roflags="0x10"|"0x10000"]`	Windows only. To ensure the consistency of the data replicated on the 2 servers, the modification of the replicated directories/files must only take place on the `PRIM` server. If changes are made on the `SECOND` server, they are notified in the module log with the identification of the process responsible so that the administrator can correct this anomaly. This is the behavior with `roflags="0x10"`. Since SafeKit 7.4.0.31, the module can also be stopped on the `SECOND` server by setting `roflags="0x10000"`. Default value: `0x10`
`[locktimeout="100"]`	Timeout in seconds for replication requests. If a request cannot be served within this timeout, the `PRIM` server becomes `ALONE`. Default value: `100` seconds
`[sendtimeout="100"]`	Since SafeKit > 7.4.0.5 Timeout in seconds for sending TCP packets to the remote node. If a packet cannot be sent within this timeout, the `PRIM` server becomes `ALONE`. Increase this value in case of low networks. Default value: `30` seconds Note In SafeKit 7.4.0.5, the default value was 12O seconds.
`[nbrei="3"]`	Number of reintegration threads running in parallel for resynchronizing files.

	Default value: `3` Note This attribute's value can be changed with a dynamic configuration.
`[namespacepolicy="0"|"1"|"3"|"4"]`	In Windows, with `namespacepolicy="1"`, zone reintegration after reboot when the module has been properly stopped is not active. To enable it in Windows, set `namespacepolicy="3"`. It activates the USN change journal on the volume containing the replicated directories (see `fsutil usn` command for creating USN change journal on a volume). Even with this configuration, full reintegration is used instead of zone reintegration when: ➩ the USN change journal associated with the volume has been deleted/recreated for administration reasons ➩ discontinuity in the USN journal is detected When zone synchronization is not possible (on the first reintegration or when zones are not available), the files that need to be synchronized are fully copied. If this reintegration does not complete, the next one will copy again these files. To avoid this, set `namespacepolicy="4"`. This option also enables USN journal checking in Windows. Set `namespacepolicy="0"` to deactivate the zone reintegration on Windows or Linux. Default value: `4` since SafeKit > 7.4.0.5 (not supported in previous releases)
`[reitimeout="150"]`	Timeout in seconds for reintegration requests. The timeout can be increased to avoid reintegration failure on heavy load of the primary server. Default value: `150` seconds Note This attribute's value can be changed with a dynamic configuration.
`[reicommit="0"]`	Linux only. Set `reicommit="nb blocks"` to commit every (nb blocks)* `reipacketsize` when reintegrating one file (in addition to the commit at the end of the copy). This can help to succeed reintegration of big files but slows down reintegration time. Default value: `0` that means no intermediate commit
`[reidetail="on"|"off"]`	Detailed logging for reintegration. Default value: `off`

	Note This attribute’s value can be changed with a dynamic configuration.
`[allocthreshold="0"]`	Windows only. Size in Gb to apply the allocation policy before reintegration. When `allocthreshold> 0`, enable fast allocation of disk space for files to be synchronized on the secondary node. This feature avoids a timeout when the primary writes at the end of the file, when the file is very large (> 200 Gb) and not yet completely copied. Since SafeKit 7.4.0.64, the allocation policy has changed and is applied for: ⇨ Newly created file or already existing empty file on the secondary and the file size on the primary is >= `allocthreshold` or ⇨ Newly created file or already existing file on the secondary and the primary file size – secondary file size is >= `allocthreshold` and the file needs full synchronization. Full synchronization is applied on the first reintegration; on start with full synchronization (`safekit second fullsync`) ; or when synchronization by zones is disabled (`namespacepolicy=”0”`) Default value: `0` (that disables the feature)
`[nbremconn="1"]`	Number of TCP connections between the primary and the secondary nodes. This value may be increased to improve the replication and synchronization throughput when the network has high latency (in cloud for instance). Default value: 1
`[checktime="220000"]`	Linux only. Timeout in milliseconds for the null request that checks the local replicated file system. Run the `safekit stopstart` command when the timeout is reached. Default value: `220 000` milliseconds
`[checkintv="120"]`	Linux only. Interval in seconds between two null requests. Default value: `120` seconds
`nfsbox_options="cross"|"nocross"`	Windows only. It specifies the policy to apply when a reparse point of type `MOUNT_POINT` is present in the replicated directory tree. This policy applies to all replicated directories.

	MOUNT_POINT reparse points in NTFS can represent two types of objects: an NTFS mount point (for example the D:\ directory) or an NTFS "directory junction" (a form of "symbolic link" to another part of the file system namespace). When nfsbox_options="cross", the MOUNT_POINT reparse point object itself is not replicated/reintegrated. It is evaluated, and the reintegration/replication process the target content as it would do for the content of a standard directory. This is useful for instance when a replicated directory is a mount point (e.g., replicating a "drive letter" root). This is the default configuration value. When nfsbox_options="nocross", the MOUNT_POINT reparse point object itself is replicated/reintegrated, but not evaluated. Reintegration does not descend into the target of the reparse point. This is useful for instance when a replicated directory tree contains NTFS "junctions" that point to another part of the replicated tree (e.g., when replicating a PostgreSQL database, as PostgreSQL is known to need such objects). Default value: cross
[scripts= "on" | "off"]	scripts="on" activates _rfs_* script callbacks used to implement external data replication management (see Linux drbd.safe module for more information) Default value: off
[reiallowedbw="20000"]	When defined, this attribute specifies the maximum bandwidth that the reintegration phase may use (for instance 20000 KB/s), in kilo bytes per second (KB/s). Due to implementation trade-off, a +/-10% fluctuation of the effectively used bandwidth is to be expected. Note The replication bandwidth is not affected by this parameter. By default, the attribute is not defined, and the bandwidth used by the reintegration is not limited
[syncdelta="nb minutes"]	When <=1, the attribute is ignored and the default failover and start policy is applied: only an up-to-date server can start as primary or run a failover. When >1, it changes the default failover and start policy. The not up-to-date server can become primary but only if the elapsed time, in minutes, since the last synchronization is lower than the syncdelta value (see 13.6.4.4 page 261). Default value: 0 minutes
[syncat="synchronization scheduling"]	Default: real-time replication and automatic synchronization (no scheduling) Use syncat for scheduling the synchronization of replicated directories on the secondary node (see 13.6.4.10 page 268). The module must be started for enabling this feature. Once

	synchronized, the module blocks in the WAIT (red) state until the next synchronization. The scheduling is based on native job scheduler: ⇨ On Unix, the job is defined in the safekit user’s crontab ⇨ On Windows, the job is defined as a system task You must configure syncat with the syntax of the native job scheduler. For instance, for synchronizing daily, after midnight: ⇨ in Windows syncat="/SC DAILY /ST 00:01:00" ⇨ in Unix syncat="01 0 * * *" Note: See crontab documentation in Unix and schtasks.exe documentation in Windows, for the full syntax of scheduled date and time. Important: Since SafeKit configuration is just a front end to the job scheduler, when scheduling is not working, please check first for syntax errors.
[<flow name =”network”> [<server addr="IP_1" /> <server addr="IP_2" />] </flow>]	Obsolete configuration preserved for backwards compatibility. When this section is not defined, the replication flow uses the same network as the heartbeat with ident="flow" if there is one, if not it uses the first heartbeat (see 13.3 page 239). If you define this section, be coherent with heartbeat ident="flow", if there is one, because default failover rules apply to this heartbeat (see 13.18.5 page 290). Note: This <flow> tag subtree can be changed with a dynamic configuration for setting a new replication flow for instance. The name attribute of <flow> define the network used for replication flow. It must present in global cluster configuration (see 12 page 227). The <server> tag is a legacy syntax used in previous SafeKit version (before 7.2). It’s supported for compatibility reason but must not be used for new modules. Important: In the same userconfig.xml, you must not use the syntax for SafeKit 7.1 and the one for SafeKit 7.2.
<replicated	Begin the definition of replicated directories. Set as many lines as there are replicated directories.

`dir="/abs_path"`	Absolute path of a directory to replicate.
`[mode= "read_only"]`	Read-only access rights on the secondary machine for replicated directories to avoid corruption
`<notreplicated path="relative" />`	Relative path of a file or sub-directory in a replicated directory. The file (or sub-directory) is not replicated. Set as many lines as there are non-replicated files or sub-directories.

`<notreplicated regexpath="regular expression" />`	Linux only.
	Regular expression to define non-replicated files or sub-directories in the replicated directory.
	Example (for more information, type "man regex"):
	`<replicated dir="/safedir">`
	`<notreplicated regexpath=".*\.tmp" />`
	`</replicated>`
	In this example, /safedir/conf/config.tmp and /safedir/log.tmp are not replicated while /safedir/conf/config.tmp.bak is replicated.

`<tocheck path="relative" />`	Relative path of a file or sub-directory in a replicated directory. Checks the presence of the file or sub-directory before starting the replication mechanism. Avoids errors such as starting replication on an empty file system. Set as many lines as there are files or sub-directories to check.

13.6.4 <rfs> description

13.6.4.1 <rfs> prerequisites

See file replication prerequisites described in 2.2.4 page 29.

13.6.4.2 <rfs> Linux

On Linux, interception of data is based on a local NFS mount. And the replication flow between servers is based on NFS v3 / TCP protocol.

The NFS mount of replicated directories from remote Unix clients is not supported. The NFS mount of other directories can be made with standard commands.

Procedure for replicating a mount point

When replicating a mount point in Linux, the module configuration fails with the error:

```
Error: Device or resource busy
```

In the following, we take the example of PostgreSQL module that set as replicated directories `/var/lib/pgsql/var` and `/var/lib/pgsql/data`. The `userconfig.xml` of the module contains:

```
<rfs … >
         <replicated dir="/var/lib/pgsql/var" mode="read_only" />
         <replicated dir="/var/lib/pgsql/data" mode="read_only" />
</rfs>
```

These directories are mount points as shown by the result of the command `df -H`. It returns for instance:

```
/dev/mapper/vg01-lv_pgs_var … /var/lib/pgsql/var
```

```
/dev/mapper/vg02-lv_pgs_data … /var/lib/pgsql/data
```

You must apply the following procedure for configuring the module to replicate these directories.

Important

It is the same procedure for all mounts points that must be replicated.

- umount the file systems by running the commands:

  ```
  umount /var/lib/pgsql/var
  umount /var/lib/pgsql/data
  ```

- configure the module by running the command:

  ```
  /opt/safekit/safekit config -m postgresql
  ```

  The configuration should succeed (no errors)

- check the symbolic links created by running the command `ls -l /var/lib`. It returns:

  ```
  lrwxrwxrwx 1 root var -> var_For_SafeKit_Replication
  lrwxrwxrwx 1 root data -> data_For_SafeKit_Replication
  ```

- edit `/etc/fstab` and change the two lines:

  ```
  /dev/mapper/vg01-lv_pgs_var /var/lib/pgsql/var ext4…
  /dev/mapper/vg02-lv_pgs_data /var/lib/pgsql/data ext4…
  ```

  with

  ```
  /dev/mapper/vg01-lv_pgs_var
  /var/lib/pgsql/var_For_SafeKit_Replication ext4…

  /dev/mapper/vg02-lv_pgs_data
  /var/lib/pgsql/data_For_SafeKit_Replication ext4..
  ```

- mount the file systems by running the commands:

  ```
  mount /var/lib/pgsql/var_For_SafeKit_Replication
  mount /var/lib/pgsql/data_For_SafeKit_Replication
  ```

Important

Apply this procedure on both nodes if replicated directories are mount point on both nodes. Once applied, you can use the module as usual: i.e., safekit start stop etc ...

Note

To protect the start of the module on a non-mounted and empty directory, you can insert in `userconfig.xml` the checking of a file inside the replicated directory. Example for `/var/lib/pgsql/var` (do the same for `/var/lib/pgsql/data` with a file inside this directory which is always present):

```
<replicated dir="/var/lib/pgsql/var" mode="read_only">
    <tocheck path="postgresql.conf" />
</replicated>
```

If you want to unconfigure the module (or uninstall whole SafeKit package), you must reverse this procedure by:

- umount the file systems with:

  ```
  umount /var/lib/pgsql/var_For_SafeKit_Replication
  umount /var/lib/pgsql/data_For_SafeKit_Replication
  ```

- de-configure the module with `/opt/safekit/safekit deconfig -m postgresql`
- edit `/etc/fstab` to undo previous editing
- mount the file systems with:

  ```
  mount /var/lib/pgsql/var
  mount /var/lib/pgsql/data
  ```

13.6.4.3 <rfs> Windows

On Windows, interception of data is based on a file system filter. And the replication flow between servers is based on NFS v3 / TCP protocol.

The <rfs> filter may not work correctly with some anti-viruses.

On Windows, you can mount remotely a replicated directory from a workstation. If you want to mount with the virtual name instead of the digital virtual IP address, you must set the two following registry keys on the server side:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa]
"DisableLoopbackCheck"=dword:00000001
```

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\lanmanserver\parameters] "DisableStrictNameChecking"=dword:00000001
```

In Windows, to enable zone reintegration after server reboot, when the module has been successfully stopped, the <rfs> component uses the NTFS USN log to verify that the information recorded on the zones is still valid after the reboot. When the control succeeds, the zone reintegration can be applied to the file; otherwise, the file must be fully copied.

By default, only the system drive has a USN log active. If the replicated directories are located on a different drive than the system drive, you must create the log (with `fsutil usn command`). See SK-0066 for an example.

13.6.4.4 <rfs> replication and failover

With its file-replication function, mirror architecture is particularly suitable for providing high availability for back-end applications with critical data to protect against failure. The reason is that the secondary server data is strongly synchronized with the primary server data. A synchronized server is considered as up-to-date and only an up-to-date server can start as primary or run a failover.

If the application availability is more critical than the application data, this default policy can be relaxed by allowing a server to become primary if the time elapsed since the last synchronization is below a configurable delay. This is configured by setting the `syncdelta` attribute of the <rfs> tag:

- `syncdelta` <= 1

  The attribute is ignored and the default failover and start policy is applied. The default value is 0.

- `syncdelta` > 1

  When the last up-to-date server is not responding, the not up-to-date server can become primary but only if the elapsed time since the last synchronization is lower than the `syncdelta` value (in minutes).

This feature is implemented with:

- `rfs.synced` resource

  When `syncdelta` is > 1, the `rfs.synced` resource is managed. This resource is `UP` if the replicated data are consistent and if the elapsed time, in minute since the last synchronization is lower than the `syncdelta` value.

- `syncedcheck` checker

  When `syncdelta` is > 1, this checker is running. It sets the value for the `rfs.synced` resource.

- `rfs_forceuptodate` failover rule

  When `syncdelta` is > 1, the following failover rule is valid:

  `rfs_forceuptodate`*:   if (heartbeat.* == down && cluster() == down && rfs.synced == up && rfs.uptodate == down) then rfs.uptodate=up;*

  This rule leads to the primary start of the server when the up-to-date server is not responding and if the server is isolated and can be considered as synchronized according to `syncdelta` value.

13.6.4.5 <rfs> replication verification

You can check for the module, named AM, that files are identical on the primary and the secondary, by running the following command on the `SECOND` server: `safekit rfsverify -m AM`. Run `safekit rfsverify -m AM > log` to redirect the command output into the file named `log`.

This output of the command is a log like that of the reintegration in which the files to be copied (therefore different) are indicated. When on the primary, there is activity on the replicated directories, an anomaly may be detected while there is no difference between the files in the following cases:

- on Windows because modifications are made on disk before being replicated
- with `async="second"` (default) because reads can bypass the asynchronous writes.

To check if there is really an inconsistency, you must re-run the command on the secondary server making sure that there is no more activity on the primary.

On Windows, some files are systematically seen as erroneous by the verifier while there is no difference. This occurs when files are modified with `SetvalidData`: files are extended without resetting the new extension and the reads return random data from the disk.

Note

It is strongly recommended to run this command only when there are no accesses to the replicated directories on the primary.

13.6.4.6 <rfs> file changes since the last synchronization

Before starting a secondary server, it may be useful to evaluate the number of files and data that have been changed on the primary server since the secondary server has stopped. This feature is provided by running the following command on the `ALONE` server: `safekit rfsdiff -m AM`. Run `safekit rfsdiff -m AM > log` to redirect the command output into the file named `log`.

This command runs on-line checks of regular files content of the module `AM`. It scans the entire replicated tree and displays the number of files that have been modified as well as the size that need to be copied. It also displays estimation for the synchronization duration. This is only estimation since only regular files are scanned and some other modifications may occur until the synchronization is run by the secondary server.

This command must be used with caution on a production server since it leads to an overhead on the server (for reading trees and files with locking). On Windows, rename of files can fail during the evaluation.

Note

It is strongly recommended to run this command only when there are no accesses to the replicated directories.

13.6.4.7 <rfs> replication and reintegration bandwidth

The replication component monitors, on the `PRIM` server, the bandwidth used by replication and reintegration write requests.

Two resources (`rfs.rep_bandwidth` and `rfs.rei_bandwidth`) reflect the average bandwidth used by replication and reintegration respectively during the last 3 seconds, expressed in kilo bytes per second (KB/s).

Note

In SafeKit < 7.5, these resources were named `rfs_bandwidth.replication` and `rfs_bandwidth.reintegration`.

If the replication load is IO intensive, the reintegration phase may saturate the network link and significantly slow down the application. In such a case, the `<rfs> reiallowedbw` attribute may be used to limit the bandwidth taken by the reintegration phase (see 13.6.3 page 252). Please note that limiting the reintegration bandwidth will make the reintegration phase longer.

Since SafeKit 7.5, there are 2 new resources that reflect the network bandwidth (in in Kbytes/sec) used between nfsbox processes, that run on each node to implement replication and reintegration:

- `rfs.netout_bandwidth` is the network output bandwidth
- `rfs.netin_bandwidth` is the network input bandwidth

You can observe the value of `rfs.netout_bandwidth` on the primary or `rfs.netin_bandwidth` on the secondary to know the modification rate at the time of observation (write, create, delete, …). The history of the resource values gives an overview of its evolution over time.

The value of the bandwidth depends on the application, system, and network activity. Its measurement is available for information purposes only.

13.6.4.8 <rfs> synchronization by date

SafeKit 7.2 offers a new command `safekit secondforce -d date -m AM` that forces the module `AM` to start as secondary after copying only files modified after the specified date.

Important

This command must be used with cautions since the synchronization will not copy files modified before the specified date. It is the administrator’s responsibility to ensure that these files are consistent and up-to-date.

The date is in the format of YYYY-MM-DD[Z] or "YYYY-MM-DD hh:mm:ss[Z]" or YYYY-MM-DDThh:mm:ss[Z], where:

- YYYY-MM-DD indicates the year, month, and day
- hh:mm:ss indicates the hours, minutes, and seconds
- Z indicates that the time is in UTC time zone; when not set the time is in local time zone

For instance:

- `safekit secondforce -d 2016-03-01 -m AM` for copying only files modified after the 1st of March 2016
- `safekit secondforce -d "2016-03-01 12:00:00" -m AM` for copying only files modified after the 1st of March 2016 at 12h, local time zone
- `safekit secondforce -d 2016-03-01T12:00:00Z -m AM` for copying only files modified after the 1st of March 2016 at 12h, UTC time zone

This command may be useful in the following case:

- the module is stopped on the primary server and a backup of the replicated data is done (on a removable drive for instance)
- the module is stopped on the secondary server and the replicated data is restored from the backup. It may be the first start-up or the repair of the secondary server.
- the module is started on the primary server that becomes `ALONE`
- the module is started on the secondary with the command `safekit secondforce -d date -m AM` where the date is the backup date

In this case, only the files modified since the backup date will be copied (full copy), instead of the full copy of all files.

Important

In Windows, the file modification date on the secondary server is changed when the file is copied by the synchronization process. Therefore, `safekit secondforce -d date -m AM`, where date is prior to the last reintegration on this server, has no interest.

13.6.4.9 <rfs> external synchronization

On the first synchronization, all replicated files are fully copied from the primary node to the secondary node. During the following synchronizations, necessary when the secondary node comes back, only zones modified, during the secondary downtime, of files that have been modified on the primary node during the secondary node downtime. When the replicated directories are voluminous, the first synchronization can take a lot of time especially if the network is slow. For this reason, since SafeKit> 7.3.0.11, SafeKit provides a new feature to synchronize a large amount of data that must be used in conjunction with a backup tool.

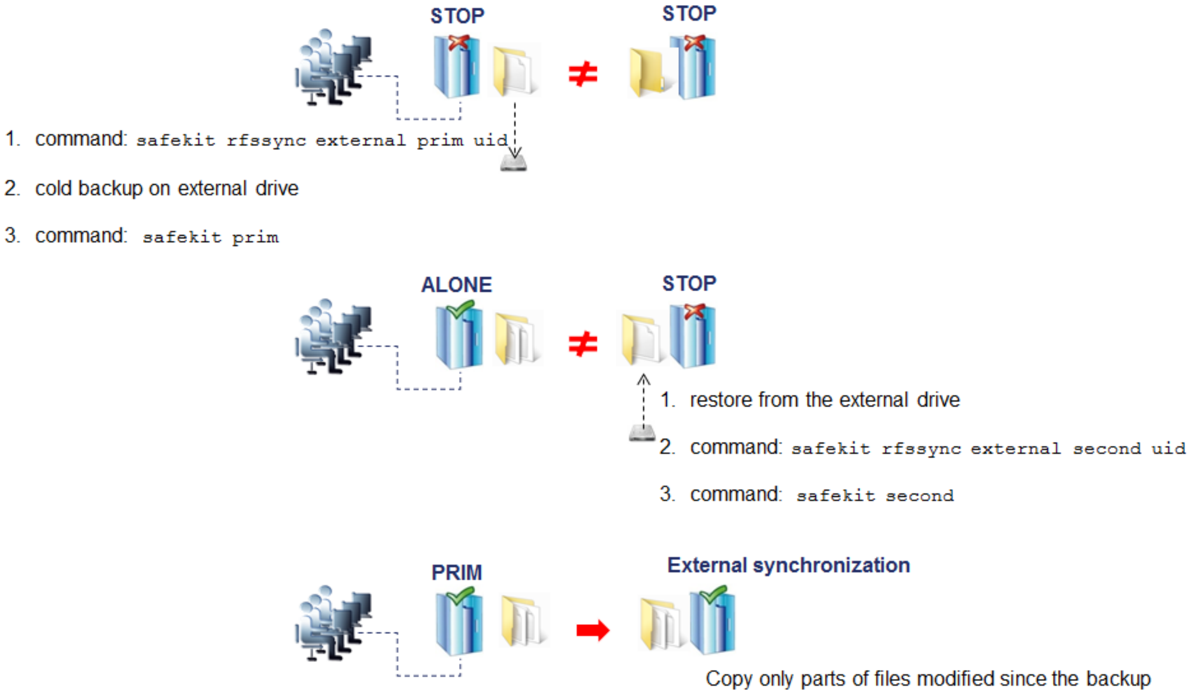
On the primary node, simply back up the replicated directories and pass the synchronization policy to the external mode. The backup is transported (using an external drive for instance) and restored to the secondary node, which is also configured to perform external synchronization. When the module is started on the secondary node, it copies only the file areas that were modified on the primary node since the backup

The external synchronization relies on a new SafeKit command `safekit rfssync` that must be applied on both nodes to set the synchronization policy to `external`. This command requires as arguments:

- the role of the node (`prim | second`)
- a unique identifier (`uid`)

External synchronization procedure

The external synchronization procedure, described below, is the procedure to be followed in the case of a cold backup of the replicated directories. In this case, the application must be stopped, and any modification of the replicated directories is prohibited until the module and the application are started, in `ALONE` - green. The order of operations must be strictly adhered to.



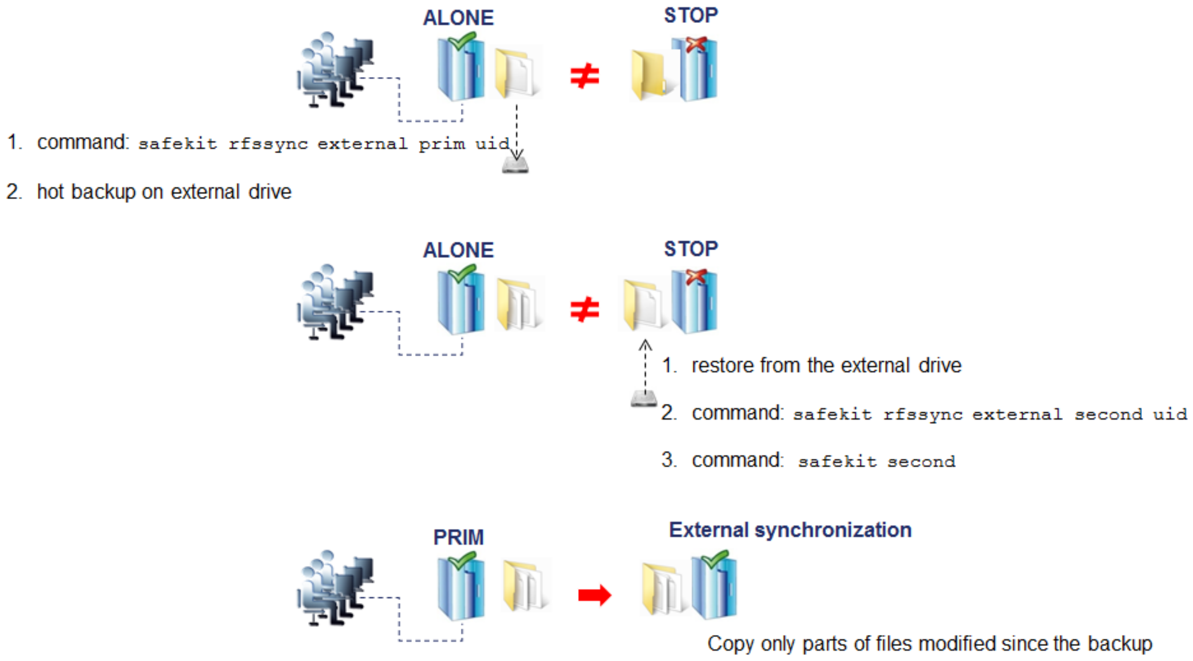
1. command: `safekit rfssync external prim uid`
2. cold backup on external drive
3. command: `safekit prim`

1. restore from the external drive
2. command: `safekit rfssync external second uid`
3. command: `safekit second`

Copy only parts of files modified since the backup

The external synchronization procedure, described below, is the procedure to be followed in the case of a hot backup of replicated directories. In this case, the module is `ALONE` -

green; the application is started and changes to the contents of the replicated directories are allowed. The order of operations must be strictly adhered to.



1. command: `safekit rfssync external prim uid`
2. hot backup on external drive

1. restore from the external drive
2. command: `safekit rfssync external second uid`
3. command: `safekit second`

Copy only parts of files modified since the backup

`safekit rfssync` command

`safekit rfssync external prim <uid> [-m AM]`	Set the synchronization policy to `external`. It is identified by the value of `uid` (at max 24 char). The node is the primary one, the source for synchronizing data.
`safekit rfssync external second <uid> [-m AM]`	Set the synchronization policy to `external`. It is identified by the value of `uid` (at max 24 char). The node is the secondary one, the destination for synchronizing data

`safekit rfssync -d prim <uid> [-m AM]`
`safekit rfssync -d second <uid> [-m AM]`
	Disable the replicated directories change detection between the cold backup/restore and the start of the module.
	Important Use this option with caution since the external synchronization may not properly detect all changes to be copied.

`safekit rfssync full [-m AM]`	Set the synchronization policy to `full`. This will copy all files in their entirety on the next synchronization.

`safekit rfssync`	Display the current synchronization policy

Internals

The synchronization policy is represented by module's resources: `usersetting.rfssyncmode`, `usersetting.rfssyncrole`, `usersetting.rfssyncuid` and `rfs.rfssync`:

- ➩ `usersetting.rfssyncmode="default"`

  `(usersetting.rfssyncrole="default", usersetting.rfssyncuid="default")`

  These values are associated with the standard synchronization policy, which is applied by default. It consists of copying only the modified areas of the files. When this policy cannot be applied, the modified files are copied in their entirety.

- ➩ `usersetting.rfssyncmode="full"`

  (`usersetting.rfssyncrole="default"`, `usersetting.rfssyncuid="default")`

  These values are associated with the `full` synchronization policy. It is applied:

  - o the first time the module is started after its first configuration
  - o on `safekit` commands (`safekit second fullsync` ; `safekit rfssync full ; safekit primforce ; safekit config ; safekit deconfig`)
  - o on change of pairing for the module

  The `full` synchronization policy will copy all files in their entirety on the next synchronization.

- ➩ `usersetting.rfssyncmode="external"`, `usersetting.rfssyncrole="prim | second"` and `usersetting.rfssyncuid="uid"`

  These values are associated with the `external` synchronization policy assigned with the commands `safekit rfssync external prim uid` and `safekit rfssync external second uid`. The next synchronization will apply the `external` synchronization policy.

- ➩ `rfs.rfssync="up | down"`

  This resource is only `up` when the synchronization policy, defined by the previous resources, can be applied.

When the synchronization policy is not the default policy, the synchronization policy automatically returns to the default mode after successful synchronization.

In some cases, external synchronization cannot be applied, and the secondary node stops with an error specified in the module log. In this situation, you must either:

- ➩ complete the external synchronization procedure if this has not been done in its entirety on the 2 nodes
- ➩ fully reapply the external synchronization procedure on the 2 nodes
- ➩ revert to the `full` synchronization policy (`safekit rfssync full` command)
- ➩ apply the synchronization by date, using the date of the backup (see 13.6.4.8 page 264). Unlike external synchronization, synchronization by date will copy the

files, modified on the primary node, in their entirety (instead of just modified parts).

13.6.4.10 <rfs> scheduled synchronization

By default, SafeKit provides real-time file replication and automatic synchronization. On heavy loaded server or high latency network, you may want to let the secondary node weakly synchronized. For this, you can use the `syncat` attribute for scheduling replicated directories synchronization on the secondary node. The module must be started for enabling this feature. Once synchronized, the module blocks in the `WAIT` (red) state until the next synchronization schedule. It is implemented with:

- the resource `rfs.syncat` that is set to `up` on the scheduled dates and set to `down` after the data synchronization
- the failover rule `rfs_syncat_wait` that blocks the module into the `WAIT` state (red) until the `rfs.syncat` resource is `up`

If you want to manually force the synchronization, you can run the command: `safekit set -r rfs.syncat -v up -m AM` while the module is in the `WAIT` (red) state.

With `syncat`, you just have to configure the scheduled time for the synchronization with the syntax of the native job scheduler: `crontab` in Linux and `schtasks.exe` in Windows (see 13.6.3 page 252).

13.7 Enable user scripts (<user>, <var> tags)

This section describes only the configuration options available for `<user>` tag. Refer to 14 page 293 for a full description of user scripts.

13.7.1 <user> example

```
<user logging="userlog" >
      <var name="VARENV" value="V1" />
</user>
```

See also the mirror module example in 15.1 page 300.

13.7.2 <user> syntax

```
<user
   [nicestoptimeout="300"]
   [forcestoptimeout="300"]
   [logging="userlog"|"none"]
   [userlogsize="2048"]
 >
      <var name="ENVIRONMENT_VARIABLE_1" value="VALUE_1" />
      …
</user>
```

Note

The `<user>` tag and full subtree can be changed with a dynamic configuration.

13.7.3 <user>, <var> attributes

`<user`	
`[nicestoptimeout="300"]`	Timeout delay in seconds to execute the `stop_xx` script. Default value: `300` seconds
`[forcestoptimeout="300"]`	Timeout delay in seconds to execute the `stop_xx` –force script. Default value: `300` seconds
`[logging="userlog"|"none"]`	stdout and stderr messages of the application started in scripts. When `logging="userlog",` messages are redirected into the log **`SAFEVAR/modules/AM/userlog.ulog`** where AM is the module name (`SAFEVAR=C:\safekit\var` on Windows and `SAFEVAR=/var/safekit` on LINUX). When `logging="none",` messages are not logged. Default value: `userlog`
`[userlogsize="2048"]`	Limit in KB of the size of the userlog On module start, the file is truncated to 0 if the size has reached this limit. Default value: `2048` KB
`<var name="ENV_VARIABLE_1" value="VALUE_1" />`	The environment variable and its value are exported before the execution of user scripts. Define as many var sections as there are environment variables to export.

13.8 Virtual hostname (<vhost>, <virtualhostname> tags)

13.8.1 <vhost> example

```
<vhost>
  <virtualhostname name="vhostname" envfile="vhostenv"/>
</vhost>
```

See also the example in 15.6 page 306.

13.8.2 <vhost> syntax

```
<vhost>
  <virtualhostname
     name="virtual_hostname"
     envfile="path_of_a_file"
    [when="prim"|"second"|"both"]
  />
</vhost>
```

Note

The <vhost> tag and subtree cannot be changed with a dynamic configuration.

13.8.3 <vhost>, <virtualhostname> attributes

<vhost>	
<virtualhostname	
name="virtual_hostname"	Definition of the virtual hostname.
envfile="path_of_envfile"	Path of the environment file automatically generated by SafeKit during configuration command If the path of the file is relative, the file will be generated in the runtime environment of the application module i.e.: SAFEUSERBIN This generated environment file is used in user scripts to set the virtual hostname before starting and stopping the application. See the module template vhost.safe delivered with Linux and Windows package.
[when="prim"|"second"|"both"]	Define when the virtual hostname must be returned to the application instead of the physical one. Default value: prim means when the server is primary (PRIM or ALONE).
/>	
</vhost>	

13.8.4 <vhost> description

Some applications need to see the same hostname on all SafeKit servers (typically, because it is stored in a replicated file). With the virtual hostname, these applications see the virtual name whereas other applications see the physical name.

See 15.6 page 306 for a complete example.

➾ On Linux

Implementation is based on the LD_PRELOAD environment variable: gethostname and uname functions are overloaded.

➾ On Windows

Implementation is based on the CLUSTER_NETWORK_NAME_ environment variable: the query API (GetComputerName, GetComputerNameEx, gethostname) functions take this variable into account. To use vhost for a service, use the command vhostservice <service> [<file>] before/after the service start/stop.

13.9 Process or service death detection (<errd>, <proc> tags)

Important

`<errd>` section requires `<user/>` section.

13.9.1 <errd> example

13.9.1.1 Process monitoring

Linux and Windows, `myproc` is the command name of the process to monitor:

```
<errd>
  <proc name="myproc" atleast="1" action="restart" class="prim"/>
</errd>
```

Linux only (since SafeKit > 7.2.0.29), `oracle_.*` is a regular expression on the command name of the process to monitor:

```
<errd>
  <proc name="oracle" nameregex="oracle_.*" atleast="1" action="restart"
class="prim"/>
</errd>
```

See also the example in 15.7 page 308.

13.9.1.2 Service monitoring

`myservice` is the name of the Windows service (since safekit > 7.3) or Linux systemd service (since safekit > 7.4.0.19) to monitor:

```
<errd>
<proc name="myservice" service="yes" atleast="1" action="restart" class="prim" />
</errd>
```

13.9.2 <errd> syntax

```
<errd
  [polltimer="10"]
>
  <proc name="command name and/or resource name for the monitored process (or
service in Windows)"
        [service="no|yes"]
        [nameregex=="regular expression on the command name"]
        [argregex="regular expression on process arguments, including command
name"]
        atleast="1"
        action="stopstart"|"restart"|"stop"|"executable_name"
        class="prim"|"both|"pre"|"second"|"sec"|"othername"]
        [start_after="nb polling cycles"]
        [atmax="-1"]
  />
  …
</errd>
```

Note

The `<errd>` tag and full subtree can be changed with a dynamic configuration.

13.9.3 <errd>, <proc> attributes

`<errd`	
`polltimer="30"`	Time delay, in seconds, between two polls of the list of processes. Default value: `30` seconds
`<proc`	Definition of a process to monitor. Set as many proc sections as there are processes. A resource is associated with each <proc>, it is named `proc.<value of the attribute name>` (e. g `proc.process_name`). The resource is `up` when the monitoring condition is true; else `down` if false.
`name="command_name"`	`name` is the command name of the process to monitor. It is also the name of the resource associated with the monitored process. At max 15 characters in Linux (the command name can be truncated); 63 in Windows. Example: on Linux, `name="vi"` and on Windows `name="notepad.exe"`. Important Windows only. The name is automatically converted to lower case. See 13.9.4 page 275 for help on retrieving the process command name.

Or
`name="command_name"`
`nameregex="regular expression on the command name"`
	Linux only
	`nameregex` is a regular expression applied on the command name for selecting the process to monitor.
	`name` is name of the resource associated with the monitored process.
	.
	Important As regular expressions are defined inside the XML file `userconfig.xml`, special characters interpreted by XML like '`<`' or '`>`' cannot be used in regular expressions.
	Example: set `nameregex="oracle _. *"` `name="oracle"` for monitoring oracle process that match the regular expression

Or
`name="service_name"`
`service="yes"`
	The associated resource is `proc.oracle`
	The `nameregex` attribute is optional
	`name` is the name of the service to monitor. It is also the name of the resource associated with the monitored service.
	At max 63 characters.
	Example:
	set `name="W32Time" service="yes"` for monitoring the Windows Time service
	set `name="ntpd" service="yes"` for monitoring the Linux Time service (systemd ntpd.service)
	The `service` attribute is optional, and the default value is `no`

`class=`
`"prim"|`
`"both"|`
`"pre"|`
`"second"|`
`"sec"|`
`"othername"`
	The process belongs to a class.
	The monitoring of a class starts only when the command `safekit errd enable "classname" -m AM` is executed.
	Activation/deactivation of `prim`, `both`, `pre`, `second`, and `sec` classes are automatically done by SafeKit in the `<user/>` component with `start_prim/stop_prim`, `start_both/stop_both`, `start_second/stop_second`, `start_sec/stop_sec`. For scripts details, see 14 page 293.
	With `another` class name, you must explicitly activate/deactivate process monitoring after/before the start/stop of the process.

`[argregex="regular expression on process arguments"]`	Regular expression matching the list of arguments of the process to monitor, including the executable name. Optional parameter.
	See 13.9.4 page 275 for help on retrieving the list of arguments of a process.
	Linux examples with vi editor on myfile ("man regex" for more information):
	`name="vi" argregex=".*myfile.*"`
	`name="vi" argregex="/myrep/myfile.*"`
	`name="vi" argregex="/myrep/myfile"`
	Windows examples with notepad editor on myfile ("class CatlRegExp" for more information):
	`name="notepad.exe" argregex=".*myfile.*"`

	name="notepad.exe" argregex="c:\\myrep\\myfile.*"
	name="notepad.exe" argregex="c:\\myrep\\myfile"
	Important As regular expressions are defined inside the XML file userconfig.xml, special characters interpreted by XML like '<' or '>' cannot be used in regular expressions.

atleast="1"	Minimum number of processes that must be running. If this minimum is not reached, then SafeKit triggers an action Example: name="oracle" argregex=".*db1.*" atleast="1" means that an action will be triggered if less than one oracle instance is running on db1. When set to -1, this criterion is meaningless. Default value: 1
action= "restart"| "stopstart"| "stop"| "noaction"| "executable_name"	Action (or handler) to execute on the application module. noaction means logging a message, restart triggers a local restart and stopstart triggers a failover. To avoid a loop on reproducible fault, a maxloop counter is incremented at each restart/stopstart command. For the maxloop definition, see section 13.2 page 236. To define a special handler, either set an absolute path or a path relative to the "bin" directory of the module: SAFE/modules/AM/bin/. We recommend a relative path and a handler defined inside the module. When defining a special handler, a new class name must be associated with the monitored process. For a special handler on Linux, on success, end with exit 0 For a special handler on Windows, on success, end with %SAFEBIN%\exitcode 0 With a different value, SafeKit performs a stopstart command. When running special handlers, the maxloop counter is not incremented. To increment it: safekit incloop -m AM -i <handler name> This command increments the counter and returns 1 when the limit has been reached. Default value: stopstart
start_after=[nb polling cycles]	Without the start_after attribute the monitoring of processes is immediately effective.

	Otherwise, it is delayed for (n-1)*polltimer (in seconds) where: ⇨ n is the value given in start_after parameter ⇨ polltimer is the value set on the errd flag (30 seconds by default) For example, if start_after="3", the server is delayed for 60 seconds ((3-1)*30). The start_after parameter is useful if the process takes a certain time to start. Default value: 0
Advanced parameters	
atmax="-1"	Maximum number of processes that can run. If this maximum is reached, then SafeKit triggers an action. atmax="-1" means that this criterion is meaningless. With atmax="0", an action is triggered each time the process is started. Default value: -1 this criterion is meaningless
</errd>	

13.9.4 <errd> commands

Note

If the command is used inside a user script, then the SAFEMODULE environment variable is set and the -m AM parameter is not necessary

safekit -r errdpoll_running	This command prints into the file **SAFEVAR/errdpoll_reserrd** (SAFEVAR=/var/safekit on Linux and SAFEVAR=c:\safekit\var on Windows if c: is the installation drive), one line for each running process with following fields: <pid> <command name> <command full name and arguments list> (parent=<parent pid>) In Windows, the command name is displayed in lower case. Useful to find the process name and its arguments for an <errd> configuration

`safekit errd disable "classname" -m AM`	Suspends the monitoring of the processes included in the class `classname` (for the application module `AM`). Must be explicitly done in stop_... scripts before stopping the application, for processes in class different from `prim`, `both`, `second`, `sec`.
`safekit errd enable "classname" -m AM`	Resumes the monitoring of the processes defined with the class `classname` (for the application module `AM`). Must be explicitly done in start_... scripts after starting the application, for processes in class different from `prim`, `both`, `second`, `sec`.
`safekit errd suspend -m AM`	Suspends the monitoring of all processes except SafeKit processes (for the application module `AM`). Useful when stopping manually the application without triggering error detection.
`safekit errd resume -m AM`	Resumes the monitoring of processes suspended with `safekit errd suspend` (for the application module `AM`).
`safekit errd list -m AM`	Lists all processes monitored by SafeKit (including SafeKit processes) and defined in the application module `AM`. The list displayed may be truncated due to internal limits. The full list can be found in the file **`SAFEVAR/modules/AM/errdlist`**. `SAFEVAR=/var/safekit` on Linux and `SAFEVAR=c:\safekit\var` on Windows if c: is the installation drive.

safekit kill -name="process_name" [-argregex="…"] -level="kill_level"	<errd> component must run. level="test": only display the process list level="terminate": kill processes level="9": send SIGKILL signal to processes (Linux only) level="15": send SIGTERM signal to processes (Linux only) Windows examples ("class CatlRegExp" for more information): safekit kill -name="notepad.exe" -argregex=".*myfile.*" -level="terminate" safekit kill -name="notepad.exe" -argregex="c:\\myrep\\myfile.*" -level="terminate" Linux examples ("man regex" for more information) : safekit kill -name="vi" -argregex=".*myfile.*" -level="9" safekit kill -name="vi" -argregex="/myrep/myfile.*" -level="9"

13.10 Checkers (<check> tag)

SafeKit brings built-in checkers with failover rules (for default failover rules details, see 13.18.5 page 290). The checkers are:

- 13.11 “TCP checker (<tcp> tags)” page 278
- 13.12 “Ping checker (<ping> tags)” page 280
- 13.13 “Interface checker (<intf> tags)” page 281
- 13.14 “IP checker (<ip> tags)” page 282
- 13.15 “Custom checker (<custom> tags)” page 283
- 13.16 “Module checker (<module> tags)” page 285
- 13.17 “Splitbrain checker (<splitbrain> tag)” page 287

13.10.1 <check> example

All built-in checkers are configured under a single <check> section:

```
<check>
  <!-- Insert below <tcp> <ping> <intf> <ip> <custom> <module> <splitbrain> tags
-->
</check>
```

13.10.2 <check> syntax

```
<check>
  <tcp …>
    <to …/>
  </tcp>
  …
  <ping …>
    <to …/>
  </ping>
  …
  <intf …>
    <to …/>
  </intf>
  …
  <ip …>
    <to …/>
  </ip>
  …
  <custom …/>
  …
  <module …>
    [<to …/>]
  </module>
…
  <splitbrain …/>
</check>
```

Note

The `<check>` tag and full subtree can be changed with a dynamic configuration.

13.11 TCP checker (<tcp> tags)

Important

By default, a <tcp> checker makes a local restart of the application when the checked tcp service is down.

13.11.1 <tcp> example

```
<check>
  <tcp ident="R1test"  when="prim" >
    <to  addr="R1" port="80"/>
  </tcp>
</check>
```

Important

Insert the <tcp> tag into the <check> section if this one is already defined.

See also example in 15.8 page 310.

13.11.2 <tcp> syntax

```
<tcp
  ident="tcp_checker_name"
  when="prim|second|both|pre"
>
  <to
   addr="IP_address" or "name_to_check"
   port="TCP_port_to_check"
   [interval="10"]
   [timeout="5"]
   />
</tcp>
```

13.11.3 <tcp> attributes

`<tcp`	Set as many `<tcp>` sections as there are TCP checkers.
`ident="tcp_checker_name"`	TCP checker name.
`when="prim|second|both"`	Use this value for a TCP checker related to the application. The `when` value sets the checker start and stop schedule respectively after and before the application's eponym start and stop scripts (`start_prim/stop_prim`, `start_second/stop_second`, `start_both/stop_both`). Action in case of failure: `safekit restart` of the application module. For default failover rules detail, see 13.18.5 page 290. At each restart, the `maxloop` counter is incremented. For its definition, see 13.2.3 page 237.
`when="pre"`	Use this value for a TCP checker not related to the application. The checker is started/stopped after/before user scripts `prestart/poststop`. You must add a special failover rule for this "tcp" checker. Typically: `external_tcp_service: if (tcp.tcp_checker_name == down) then wait();` This rule executes a stopwait and puts the application module in the `WAIT` state while the external TCP service is not responding. See 13.18 page 288 for more information. At each stopwait, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).
`<to`	
`addr="IP_@" or "name"`	IP address or name to check (ex.: `127.0.0.1` for a local service). IPv4 or IPv6 address.

`port="value"`	TCP port to check.
`[interval="10"]`	Interval in seconds between two connections trials. Default value: `10` seconds
`[timeout="5"]`	Connection establishment timeout in seconds. Default value: `5` seconds
`</tcp>`	

13.12 Ping checker (<ping> tags)

Important

By default, a <ping> checker stops the module and waits for the ping to be up.

13.12.1 <ping> example

```
<check>
  <ping ident="testR2" >
    <to addr="R2"/>
  </ping>
</check>
```

Important

Insert the <ping> tag into the <check> section if this one is already defined.

See also the example in 15.9 page 310.

13.12.2 <ping> syntax

```
  <ping
    ident="ping_checker_name"
    [when="pre"]
  >
    <to
     addr="IP_address" or "name_to_check"
     [interval="10"]
     [timeout="5"]
     />
  </ping>
```

13.12.3 <ping> attributes

`<ping`	Set as many ping sections as there are ping checkers.
`ident="ping_checker_name"`	Ping checker name as displayed in the command `safekit state -v -m AM`. Name of checkers must be unique.

[when="pre"]	Default if not set. Started/stopped after/before user scripts prestart/poststop. Executes a stopwait and puts the application module in the WAIT state if there is no reply to the ICMP ping requests (see default failover rules definition in 13.18.5 page 290). At each stopwait, the maxloop counter is incremented (see 13.2.3 page 237 for its definition).
<to	
addr="IP_@ or name"	External IP address or name to check. IPv4 or IPv6 address.
[interval="10"]	Interval in seconds between two ping requests. Default value: 10 seconds
[timeout="5"]	Reply timeout in seconds to the ping. Default value: 5 seconds
</ping>	

13.13 Interface checker (<intf> tags)

Important

By default, a <intf> checker stops the module and waits for the network interface to come back up.

13.13.1 <intf> example

```
<check>
  <intf  ident="test_eth0">
    <to  local_addr="192.168.1.10"/>
  </intf>
</check>
```

Important

Insert the <intf> tag into the <check> section if this one is already defined.

See also the example in 15.10 page 310.

13.13.2 <intf> syntax

```
  <intf
    ident="intf_checker_name"
    [when="pre"]
       >
    <to
```

```
     local_addr="interface_physical_IP_address"/>
  </intf>
```

13.13.3 <intf> attributes

`<intf`	Note <intf> sections are automatically generated on network interface when `<interface check="on">` is set (see the virtual IP definition in 13.5 page 243).
`ident="intf_checker_name"`	Interface checker name
`[when="pre"]`	Default. Started/stopped after/before user scripts `prestart/poststop`. Execute a stopwait and put the application module in the `WAIT` state if `intf` is "down" (see the default failover rules in 13.18.5 page 290). At each stopwait, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).
`<to local_addr="IP_@ />`	Physical IP address configured on the network interface to check. IPv4 or IPv6 address.
`</intf>`	

13.14 IP checker (<ip> tags)

In LINUX and Windows, this checker checks that the IP address is locally defined; in Windows it also detects IP conflicts.

Important

By default, a <ip> checker makes a local stopstart of the module when the checked ip address is down.

13.14.1 <ip> example

```
<check>
  <ip  ident="ip_check" >
    <to  addr="192.168.1.10" />
  </ip>
</check>
```

Important

Insert the <ip> tag into the <check> section if this one is already defined.

See also the example in 15.11 page 311.

13.14.2 <ip> syntax

```
  <ip
    ident="ip_checker_name"
    [when="prim"]
  >
    <to
     addr="IP_address" or "name_to_check"
     [interval="10"]
     />
  </ip>
```

13.14.3 <ip> attributes

`<ip`	Set as many ip sections as there are ip checkers.
`ident="ip_checker_name"`	ip checker name as displayed in the `safekit state -v -m AM` command. Name of checkers must be unique.
`[when="prim"]`	Default if not set. The checker is started/stopped after/before the user scripts `start_prim/stop_prim`. Action in case of failure: `safekit stopstart` of the application module (see the default failover rules in 13.18.5 page 290). At each stopstart, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).
`<to`	
`addr="IP_@ or name"`	Local IP address or name to check. IPv4 or IPv6 address.
`[interval="10"]`	Interval in seconds between two checks. Default value: `10` seconds
`</ip>`	

13.15 Custom checker (<custom> tags)

A custom checker is a program (script or other) that you develop for your module. It is a loop performing a test at an appropriate periodicity. According to the result of the test, the program sets the state of a resource ("up" or "down"). Then a special failover rule decides which action must be taken when the resource is down.

13.15.1 <custom> example

```
<check>
  <custom ident="AppChecker" when="prim" exec="mychecker"/>
</check>
```

Important

Insert the <custom> tag into the <check> section if this one is already defined. Moreover, define the customized checker as well as the associated failover rule.

See the example in 15.12 page 312.

13.15.2 <custom> syntax

```
<custom
   ident="custom_checker_name"
   when="pre|prim|second|both"
   exec="executable_path"
   arg="executable_arguments" />
```

13.15.3 <custom> attributes

`<custom`	Set as many custom sections as there are custom checkers.
`ident="custom_checker_name"`	Custom checker name (network IP address). A custom checker must set its associated resource state itself, using the command `safekit set -r custom.custom_checker_name -v up|down`.
`when="pre"`	The checker is started/stopped after/before user scripts `prestart/poststop`. You must add a special failover rule associated with the custom checker's resource. Typically: `wait_custom_checker: if (custom.custom_checker_name == down) then wait();` This rule executes a stopwait and puts the application module in the `WAIT` state while the resource is down. Note that SafeKit automatically initializes the state of the associated resource to "`init`", and the failover machine stays in the `WAIT` state as long as the state of the custom checker is not evaluated to "`up`" or "`down`". For more information on the failover machine, see 13.18 page 288. At each stopwait, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).
`when="prim"|"second"|"both"`	The checker is started/stopped after/before user scripts `start_prim/stop_prim`, `start_second/stop_second`, `start_both/stop_both`. You must add a special failover rule associated with the custom checker's resource. Typically: `restart_custom_checker: if (custom.custom_checker_name == down) then restart();` See 13.18 page 288 for more information. At each restart, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).

`exec="executable_path"`	Defines the executable path of the custom checker. Can be a binary executable or a script file. When the path of `executable_path` is relative, it is relative to `SAFEUSERBIN`. In this case, put your executable file in `SAFE/modules/AM/bin/` of your application module and use a relative path. See 10.1 page 153 for more information on path values. We recommend a relative path and an executable inside the module. In Windows, the executable can be a binary or a ps1, vbs or cmd script In Linux, the executable can be a binary or a shell script
`arg="executable_arguments"`	Defines the executable arguments when the custom checker is started.

13.16 Module checker (<module> tags)

The module checker checks the availability of another module. It is started/stopped in the `prestart` /`poststop` phase before the start of the application. When the module checker detects that the external module is `down`, SafeKit executes a stopwait and puts the server in the `WAIT` state until the external module is detected as `up` by the module checker. The module checker also triggers a stopstart when it detects that the external module is stopping or has been restarted (either by a SafeKit stopstart, restart or failover). See 13.18.5 page 290 for the default failover rules.

At each stopwait or stopstart, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).

The module checker connects to the SafeKit web service on the node running the module to get the module state (see 10.6 page 167 for details on the web service).

13.16.1 <module> example

Example for the default configuration of the SafeKit web service (protocol: HTTP, port: 9010):

```
<check>
  <module name="mirror">
    <to  addr="M1host"/>
  </module>
</check>
```

Example for the secured configuration of the SafeKit web service (protocol: HTTPS, port: 9453):

```
<check>
  <module name="mirror">
    <to  addr="M1host" port="9453" secure="on"/>
  </module>
</check>
```

Important

Insert the <module> tag into the <check> section if this one is already defined.

For examples, see 15.3 page 303 and 15.13 page 314.

13.16.2 <module> syntax

```
  <module
    [ident="module_checker_name"]
    name="external_module_name">
    [<to
     addr=" IP_@ or name the Safekit server running the external module"
     [port=port of the SafeKit httpd server"]
     [interval="10"]
     [timeout="5"]
     [secure="on"|"off"]
     />]
  </module>
```

13.16.3 <module> attributes

`<module`	Set as many `<module>` sections as there are module checkers.
`name="external_module_name"]`	Name of the module checker.
`[ident="module_checker_name"]`	Name of the external SafeKit module to check. Default: `external_module_name_<IP_@ or name of the server>`
`[<to`	Definition of the server(s) running the external module to check. Default is the local server.
`addr="IP_@ or name of the server"`	IP address or name of the external module. IPv4 or IPv6 address.
`[port=port of the SafeKit web service"]`	Port of the SafeKit web service. Default: `9010`
`[interval="10"]`	Interval in seconds between two checks. Default value: `10` seconds.
`[timeout="5"]`	Check reply timeout in seconds. Default value: `5` seconds
`[secure="on"|"off"]`	Use HTTP protocol (`secure="off"`) or HTTPS (`secure="on"`) Default value: `off`

/>]	
</module>	

13.17 Splitbrain checker (<splitbrain> tag)

SafeKit provides a splitbrain checker that is suits mirror architectures. Split brain is a situation where, due to temporary failure of all network links between SafeKit nodes, and possibly due to software or human error, both nodes switched to the primary role while isolated. This is a potentially harmful state, as it implies that the application is running on both nodes. Moreover, when file replication is enabled, modifications to the data are made on the two nodes.

The split-brain checker detects the loss of all connectivity between nodes and selects only one node to become the primary. The other node is not up-to-date anymore and goes into the `WAIT` state until:

- the heartbeat becomes available again

or

- the administrator runs `safekit` commands to force the start as primary (`safekit stop` then `safekit prim`).

The primary node election is based on the ping of an IP address, called the **witness**. The network topology must be designed so that only one node can ping the witness in case of split brain. If this is not the case, both nodes will go primary.

Important

Ping between nodes and witness must be enabled.

13.17.1 <splitbrain> example

```
<check>
  <splitbrain ident="SBtest" exec="ping" arg="192.168.1.100"/>
</check>
```

Important

Insert the <splitbrain> tag into the <check> section if this one is already defined.

13.17.2 <splitbrain> syntax

```
  <splitbrain
     ident="witness"
     exec="ping"
     arg=" witness IP address "
  />
```

13.17.3 <splitbrain> attributes

<splitbrain	Set only one splitbrain checker.

`ident="witness"`	Name displayed in the `safekit state -v -m AM` command for the witness state.
`[when="pre"]`	Fixed value. Started/stopped after/before user scripts prestart/poststop. The witness state is stored in `splitbrain.witness`. It can be displayed using the `safekit state -v -m AM` command. On splitbrain detection, the server with `splitbrain.witness=”up”` goes primary; the other one with `splitbrain.witness=”down”` sets the resource `splitbrain.uptodate` to `down` and goes into the `WAIT` state (for default failover rules, see 13.18.5 page 290). At each stopwait, the `maxloop` counter is incremented (see 13.2.3 page 237 for its definition).
`exec="ping"`	Fixed value. Use a pinger to ping the witness and set `splitbrain.witness` state.
`arg="IP_@ or name"`	External IP address or name for the witness to ping. IPv4 or IPv6 address.
`</splitbrain>`	

13.18 Failover machine (<failover> tag)

SafeKit comes with checkers (network interface, ping, TCP, custom, module checkers) which regularly (by default every 10 seconds) check resources and set the state to `up` or `down` (see 13.10 page 277 for checkers definition). The failover machine regularly (by default every 5 seconds) evaluates the global state of all resources and triggers a failover according to failover rules programmed in a simple language.

In farm architecture, the failover machine can work only on the states of local resources whereas in mirror architecture, the failover machine can work on the states of local and remote resources. As the states of resources are exchanged on heartbeat channels, it is better to have several heartbeat channels (see 13.3 page 239 for heartbeats definition).

13.18.1 <failover> example

```
<failover>
  <![CDATA[
    ping_failure: if (ping.testR2 == down) then stopstart();
  ]]>
</failover>
```

13.18.2 <failover> syntax

```
<failover [extends="yes"] [period="5000"] [handle_time="15000"]>
<![CDATA[
  label: if (expression) then action;
  …
]]>
</failover>
```

Note

The `<failover>` tag and subtree cannot be changed with a dynamic configuration.

13.18.3 <failover> attributes

`<failover`	
`[extends="yes"|"no"]`	If set to `yes`, the new failover rules extend the default failover rules (see 13.18.5 page 290 for its definition). If set to `"no"`, the new failover rules overwrite the default one (avoid this configuration). Default value: `yes`.
`[period="5000"]`	Period in milliseconds between two evaluations of failover rules. Default value: `5000` milliseconds (5 seconds)
`[handle_time="15000"]`	A failover action must be stable (the same) at least during the time `handle_time` (in milliseconds) before being applied by the failover machine. Default value: `15000` milliseconds (15 seconds). `handle_time` must be a multiple of the `period` value.

13.18.4 <failover> commands

`safekit set [-m AM] -r resource_class.resource_id -v resource_state`
`[-n] [-l]`
	This command sets the state of one resource:
	Examples:
	`safekit set -r custom.myresource -v up`
	`safekit set -r custom.myresource -v down`
	Since SafeKit 7.5, each assignment of the main resources is stored in a log to keep track of their status. Use `-n` to disable this logging or `-l` to force it.

`safekit stopwait -i "identity"`	Equivalent to `wait()` command of the failover machine (see 13.18 page 288). With stopwait, (1) `poststop` and `prestart` scripts are not executed and (2) checkers `when="pre"` are not stopped.

The other commands (`restart(), stopstart(), stop(), swap()`) of the failover machine are equivalent to control commands (with the `-i identity` parameter) described in 9.4 page 146.

Important

`maxloop / loop_interval / automatic_reboot` are applied if `-i identity` is passed to commands (for these attributes details, see 13.2 page 236). This is the case when called from the failover machine.

13.18.5 Failover rules

The default failover rules for the SafeKit checkers are:

```
<failover>
<![CDATA[
/* rule for module checkers */
module_failure: if (module.? == down) then wait();
/* rule for interface checkers */
interface_failure: if (intf.? == down) then wait();
/* rule for ping checkers */
ping_failure: if (ping.? == down) then wait();
/* rule for tcp checkers */
tcp_failure: if (tcp.? == down) then restart();
/* rule for ip checkers */
ip_failure: if (ip.? == down) then stopstart();
/* rules for splitbrain */
splitbrain_failure: if (splitbrain.uptodate == down) then wait();
]]>
</failover>
```

They are defined into **`SAFE/private/conf/include/failover.xml`**.

There are also failover rules dedicated to file replication management.

The `WAKEUP` command is automatically generated when no `wait()` rule applies.

Note

Since SafeKit 7.5, default failover rules are using a new syntax, and rules for the rfs component are set into the file **`SAFE/private/conf/include/rfs.xml`**.

In addition to the default rules, the user can define his own rules (for a custom checker for example) using the following syntax:

```
label: if ( expression ) then action;
```

with:

- `label ::= string`
- `action ::= stop() | stopstart() | wait() | restart() | swap()`

-
  ```
  expression ::= ( expression )
   | ! expression
   | expression && expression
   | expression || expression
   | expression == expression
   | expression != expression
  ```

```
      | resource ::= [local. | remote.] 0/1resource_class.resource_id
      | resource_state
```

The syntax to design the resources is as follows:

```
resource ::= [local. | remote.] 0/1resource_class.resource_id   (default: local)
resource_class ::= ping | intf | tcp | custom | module | heartbeat | rfs
resource_id ::= * | ? | name
resource_state ::= init | down | up | unknown
```

`init`	Special initialization state of a resource when the checker is not started. If a resource in the `init` state is used in a failover rule, SafeKit does evaluate the rule.
`up`	Resource OK.
`down`	Resource KO.
`unknown`	Special state of a remote resource; the remote state is unknown at the test time (ex.: when the remote module is stopped).

14. User application scripts for the module configuration

To enable user scripts call, `<user>` tag must be defined in `userconfig.xml` as described in 13.7 page 268. This tag could be added or removed dynamically.

Scripts must executables:

- ✓ in Windows, an executable with the extension and type: `.cmd`, `.vbs`, `.ps1`, `.bat` or `.exe`
- ✓ in Linux, any type of executable

Each time you update scripts, you must apply the module configuration onto the servers (with the SafeKit console or command).

Examples of scripts are given in 15.1 page 300 for a mirror module, and in 15.2 page 301 for a farm module.

Note

During the configuration phase, scripts are copied from `SAFE/modules/AM/bin` in the execution environment directory `SAFE/private/modules/AM/bin` (=`SAFEUSERBIN`, do not touch scripts at this place) where AM is the module name.

14.1 List of scripts

Below the list of scripts that can be defined by the user. The essential scripts start/stop are those that start and stop the application within the module.

14.1.1 Start/stop scripts

`start_prim` `stop_prim`	**Scripts for a mirror module.** To start & stop application on the `ALONE` or `PRIM` server
`start_both` `stop_both`	**Scripts for a farm module.** To start & stop application on all `UP` servers in a farm cluster In the special case they are defined in a mirror module, they are also executed on both servers (`PRIM`, `SECOND` or `ALONE`)
`start_second` `stop_second`	**Special scripts for a mirror module** To start & stop application on the "`SECOND`" server Note When the secondary server becomes the primary one, `stop_second` followed by `start_prim` is executed

start_sec stop_sec	**Special scripts for a mirror module**
stop_[both, prim, second, sec] force	**Scripts for all modules** The stop scripts are called twice: once for a graceful shutdown of the application (without force as first argument), a second time with a force parameter for a rapid shutdown (with force as first argument).
prestart poststop	**Scripts for all modules** Executed at the very beginning of the module start and at its end. By default, prestart contains stop_sec, stop_second, stop_prim, stop_both to stop application before starting the module under the control of SafeKit.
transition	**Script for all modules** This script is executed on state transitions described in 14.2 page 295

14.1.2 Other scripts

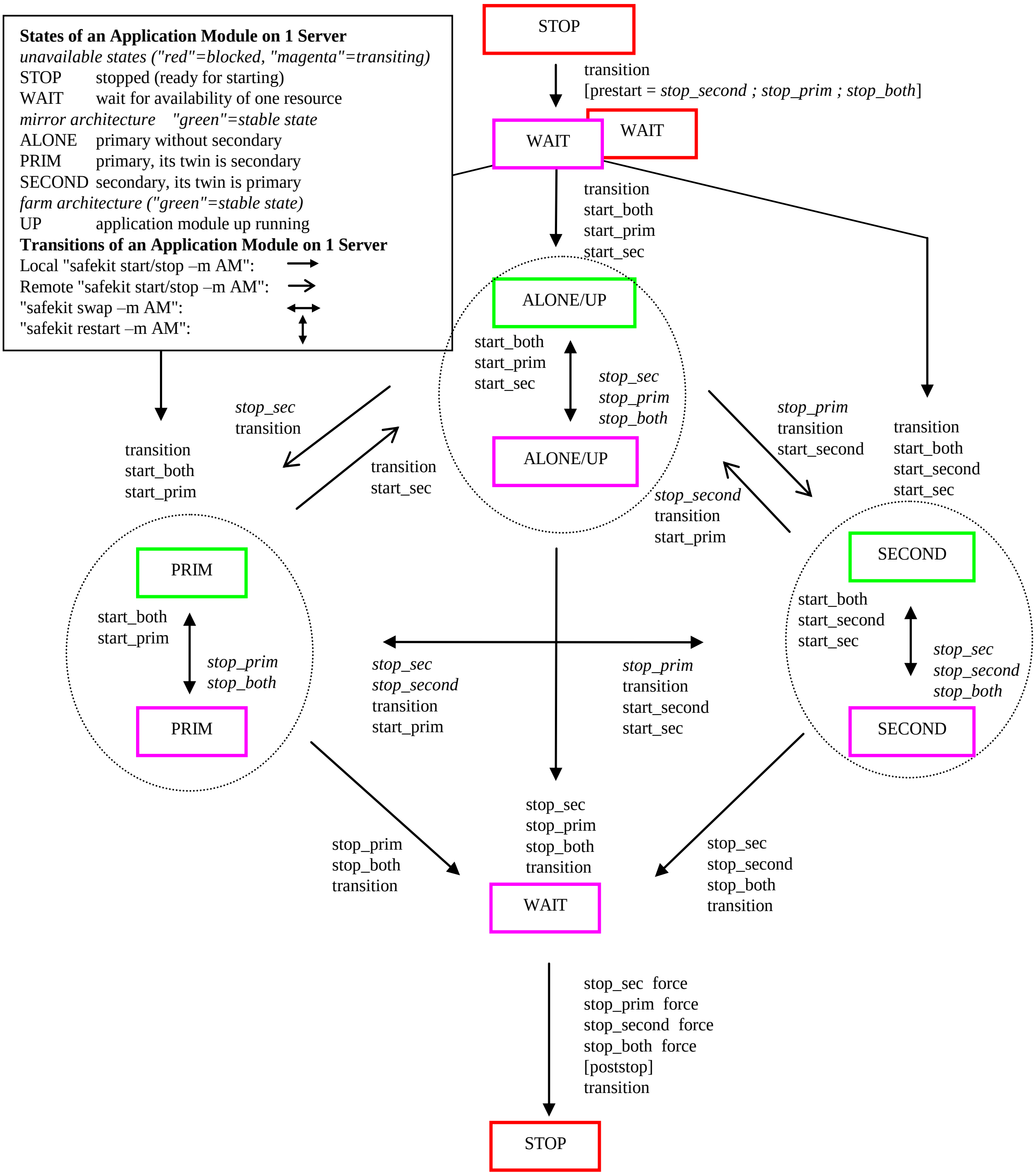
config	config is called when executing the safekit config -m AM command on the application module. You can make a special application configuration in this script.
deconfig	deconfig is called when executing the safekit deconfig -m AM command, which is itself called at the application module uninstallation. You can remove a special application configuration made previously in the config script.
confcheck	confcheck is called when executing the safekit confcheck -m AM command on the application module. You can add in this script some tests for checking changes on the application configuration files.
state	state is called when executing the safekit state -v -m AM command on the application module. You can display a special state of the application.
level	level is called when executing the safekit level -m AM command on the application module. You can display the application version.

14.2 Script execution automaton

Important

Example: first transition from `STOP` to `WAIT` calls the script `transition STOP WAIT start` is called.

Most of the time, stop scripts are called twice (without the `force` parameter and then with the `force` parameter). In that case the script name is written in italic.



14.3 Variables and arguments passed to scripts

All scripts are called with 3 parameters:

- the current state (`STOP,WAIT,ALONE,PRIM,SECOND,UP`),
- the next state (`STOP,WAIT,ALONE,PRIM,SECOND,UP`)
- the action (`start`, `stop`, `stopstart` or `stopwait`).

The stop scripts are called twice:

- a first time for a graceful shutdown of the application
- a second time with a `force` parameter for a forced shutdown (with `force` as first argument)

The environment variables that can be used inside scripts are:

- `SAFE, SAFEMODULE, SAFEBIN, SAFEUSERBIN, SAFEVAR, SAFEUSERVAR` (for details, see 10.1 page 153)
- all variables defined in `<user>` tag of `userconfig.xml` (see 13.7 page 268).

14.4 SafeKit special commands for user scripts

Special commands are installed under `SAFE/private/bin`. Special commands can be called directly in user scripts with `%SAFEBIN%\specialcommand` or `$SAFEBIN/specialcommand`. Outside user scripts, use `safekit -r` command.

`safekit -r <special command> [<args>]`	<special command> <args> executed within the SafeKit environment. When the command name is not an absolute path, the command is searched in `SAFEBIN=SAFE/private/bin` directory. Note If you use special commands outside SafeKit scripts, prefix them with `safekit -r specialcommand`

14.4.1 Commands for Windows

14.4.1.1 sleep, exitcode, sync commands

On Windows, you can use the following basic commands:

- `%SAFEBIN%\sleep.exe <timeout value in seconds>`

  To be used inside stop scripts because net stop service is not synchronous

- `%SAFEBIN%\exitcode.exe <exit value>`

  To return an error value when the script exits

- `%SAFEBIN%\sync.exe \\.\<drive letter:>`

  To sync file system cache of a disk

14.4.1.2 namealias command

➾ `%SAFEBIN%/namealias [-n | -s ] <alias name>`

`-n` to add a new NetBIOS name (set into `start_prim`) or `-s` to suppress the NetBIOS name (set into `stop_prim`)

You can also use the SafeKit command `netnames` (or the windows command `nbtstat`) to list NetBIOS information.

14.4.2 Commands for Linux

14.4.2.1 Managing the crontab

`$SAFEBIN/gencron`	
`[del | add]`	`del` to disable the entries in `stop_prim` (by inserting comments) or `add` to enable the entries in `start_prim` (by removing comments).
`<user name>`	User name in the crontab.
`[all |<command name>]`	`all`: to apply on all entries or to apply on the name of the command
`-c "<comment>"`	Header of the comment that will be inserted.

For example, to disable/enable the entry from the admin's crontab,

```
5 0 * * * $HOME/bin/daily.job >> $HOME/tmp/out 2>&1
```

Insert into `stop_prim`:

```
$SAFEBIN/gencron del admin daily.job -c "SafeKit configuration for
$SAFEMODULE"
```

And insert into `start_prim`:

```
$SAFEBIN/gencron add admin daily.job -c "SafeKit configuration for
$SAFEMODULE"
```

14.4.2.2 Bounding command

```
$SAFEBIN/boundcmd <timeout value> <command path> [<args>]
```

Bound a command with a timeout

`boundcmd` returns the exit code of the command when the command terminates before the timeout; otherwise, it exits with the value 2.

For example, to flush data on disk with a timeout of 30 seconds, run:

```
$SAFEBIN/boundcmd 30 /bin/sync 1>/dev/null 2>&1
```

14.4.2.3 Commands for Windows and Linux

<table>
<tr><td>safekit -r processtree list | kill …</td><td>List running processes as a tree (except for all) and optional kill
⇨ safekit -r processtree list all
List all running processes.
⇨ safekit -r processtree list <process command name>
List all running processes with the specified command name.
⇨ safekit -r processtree kill <process command name>
List and kill all running processes with the specified command name.
⇨ safekit -r processtree list | kill <process command name>| all <regular expression on the full command –path and arguments>
List (and kill) all running process with the specified command name and arguments.
Windows examples ("class CatlRegExp" for more information):
safekit -r processtree kill notepad.exe ".*myfile.*"
safekit -r processtree list all “mirror”
Linux examples ("man regex" for more information) :
safekit -r processtree kill vi ".*myfile.*"
safekit -r processtree list all “mirror”</td></tr>
<tr><td>safekit incloop
-m AM -i <handler name&gt;</td><td>SafeKit provides a maxloop counter, the number of restart and stopstart of the module on error detection. The module is stopped when this counter reaches the maxloop value over the loop_interval period.
When running special handlers, the maxloop counter is not incremented. To increment it, use the command:
safekit incloop -m AM -i <handler name>
It increments the maxloop counter for the module AM and returns 1 when the limit has been reached.</td></tr>
<tr><td>safekit resetloop
-m AM [-i <handler name&gt;]</td><td>Reset the maxloop counter to the value 0</td></tr>
<tr><td>safekit checkloop -m AM</td><td>For checking the maxloop counter for the module AM, use the command: safekit checkloop -m AM
⇨ It returns 0 when the maxloop counter is not reached or the last increment occurred outside loop_interval
⇨ It returns 1 when the maxloop counter is reached and the last increment occurred during loop_interval</td></tr>
</table>

15. Examples of userconfig.xml and user scripts

Some examples are taken from the modules delivered with the SafeKit package, under `SAFE/Application_Modules`. You can install them with the web console (see 3.7.4 page 62) to examine the configuration file and user scripts in detail.

Other examples of integration are described under https://www.evidian.com/products/high-availability-software-for-application-clustering/cluster-configuration/.

Important

The .safe are platform dependent and therefore different in Windows and Linux.

In the following, the examples use this global cluster configuration:

```
<cluster>
   <lans>
      <lan name="net3">
         <node name="node1" addr="10.1.0.2"/>
         <node name="node2" addr="10.1.0.3"/>
         <node name="node3" addr="10.1.0.3"/>
      </lan>
      <lan name="default" console="off">
         <node name="node1" addr="192.168.1.1"/>
         <node name="node2" addr="192.168.1.2"/>
      </lan>
      <lan name="repli" console="off">
         <node name="node1" addr="10.0.0.2"/>
         <node name="node2" addr="10.0.0.3"/>
      </lan>
   </lans>
</cluster>
```

15.1 Generic mirror module example with `mirror.safe`

Below is the configuration file and user scripts of the generic mirror module, `mirror.safe`, in Windows. For Linux, please refer to the `mirror.safe` delivered with the Linux package.

`conf/serconfig.xml` - see 13 page 235

```
<!-- Mirror Architecture with Real Time File Replication and Failover -->
<!DOCTYPE safe>
<safe>
   <service mode="mirror" defaultprim="alone" maxloop="3" loop_interval="24"
failover="on">
      <heart pulse="700" timeout="30000">
         <heartbeat name=”default” ident=”flow”/>
      </heart>
      <rfs async="second" acl="off" locktimeout="100" nbrei="3" iotimeout="300">
         <replicated dir="c:\test1replicated" mode="read_only"/>
         <replicated dir="c:\test2replicated" mode="read_only"/>
      </rfs>
      <vip>
         <interface_list>
            <interface check="on" arpreroute="on">
               <real_interface>
                  <virtual_addr addr="192.168.4.10" where="one_side_alias"/>
               </real_interface>
            </interface>
         </interface_list>
      </vip>
      <user nicestoptimeout="300" forcestoptimeout="300" logging="userlog"/>
   </service>
</safe>
```

`bin/start_prim.cmd` - see 14 page 293

```
@echo off
rem Script called on the primary server for starting application services
rem For logging into SafeKit log use:
rem "%SAFE%\safekit" printi | printe "message"

rem stdout goes into Application log
echo "Running start_prim %*"
set res=0

rem Fill with your services start call
rem net start "myservice" /Y

set res=%errorlevel%

if %res% == 0 goto end

:stop
"%SAFE%\safekit" printe "start_prim failed"
rem uncomment to stop SafeKit when critical
rem "%SAFE%\safekit" stop -i "start_prim"

:end
```

`bin/stop_prim.cmd` - see 14 page 293

```
@echo off
rem Script called on the primary server for stopping application services
rem For logging into SafeKit log use:
rem "%SAFE%\safekit" printi | printe "message"

rem ----------------------------------------------------------
rem
rem 2 stop modes:
rem
rem - graceful stop
rem   call standard application stop with net stop
rem
rem - force stop (%1=force)
rem   kill application's processes
rem
rem ----------------------------------------------------------

rem stdout goes into Application log
echo "Running stop_prim %*"

set res=0

rem default: no action on forcestop
if "%1" == "force" goto end

rem Fill with your service(s) stop call
rem net stop "myservice" /Y

rem If necessary, uncomment to wait for the stop of the services
rem "%SAFEBIN%\sleep" 10

if %res% == 0 goto end

"%SAFE%\safekit" printe "stop_prim failed"

:end
```

15.2 Generic farm module example with `farm.safe`

Below is the configuration file and user scripts for the generic farm module, `farm.safe`, in Windows. For Linux, please refer to the `farm.safe` delivered with the Linux package.

`conf/userconfig.xml` - see 13 page 235

```
<!-- Farm Architecture with Load-Balancing and Failover -->
<!DOCTYPE safe>
<safe>
   <service mode="farm" maxloop="3" loop_interval="24">
      <!-- Cluster Configuration -->
      <!-- Set nodes on your network -->
      <farm>
         <lan name=”default” />
         <lan name =”net3” />
      </farm>
      <vip>
         <interface_list>
            <interface check="on" arpreroute=”on”>
```

```
               <virtual_interface type="vmac_directed">
                  <virtual_addr addr="192.168.4.20" where="alias"/>
               </virtual_interface>
            </interface>
         </interface_list>
         <loadbalancing_list>
            <group name="FarmProto">
               <!-- Set load-balancing rule -->
               <rule port="9010" proto="tcp" filter="on_port"/>
            </group>
         </loadbalancing_list>
      </vip>
      <user nicestoptimeout="300" forcestoptimeout="300" logging="userlog"/>
   </service>
</safe>
```

`bin/start_both.cmd` - see 14 page 293

```
@echo off

rem Script called on all servers for starting applications

rem For logging into SafeKit log use:
rem "%SAFE%\safekit" printi | printe "message"

rem stdout goes into Application log
echo "Running start_both %*"

set res=0

rem Fill with your services start call
rem net start "myservice" /Y

set res=%errorlevel%

if %res% == 0 goto end

:stop
set res=%errorlevel%
"%SAFE%\safekit" printe "start_both failed"

rem uncomment to stop SafeKit when critical
rem "%SAFE%\safekit" stop -i "start_both"

:end
```

`bin/stop_both.cmd` - see 14 page 293

```
@echo off

rem Script called on all servers for stopping application

rem For logging into SafeKit log use:
rem "%SAFE%\safekit" printi | printe "message"

rem ----------------------------------------------------------
rem
rem 2 stop modes:
```

```
rem
rem - graceful stop
rem   call standard application stop with net stop
rem
rem - force stop (%1=force)
rem   kill application's processes
rem
rem ----------------------------------------------------------

rem stdout goes into Application log
echo "Running stop_both %*"

set res=0

rem default: no action on forcestop
if "%1" == "force" goto end

rem Fill with your services stop call
rem net stop "myservice" /Y

rem If necessary, uncomment to wait for the stop of the services
rem "%SAFEBIN%\sleep" 10

if %res% == 0 goto end

"%SAFE%\safekit" printe "stop_both failed"

:end
```

15.3 A Farm module depending on a mirror module example

In the example below, the farm module can only start if the mirror module is started. This architecture can be used to link an IIS farm module to a Microsoft SQL server mirror module. It is based on the configuration of a module checker in the farm module. For details, see 13.16 page 285.

farm/conf/userconfig.xml - see 13 page 235

```
…
  <!-- Checker Configuration: module dependency to mirror + local TCP checker -->
  <check>
    <module name="mirror">
     <to addr="192.168.1.31"/>
    </module>
   </check>
…
```

Note

Note that the module dependency can be used when you deploy farm and mirror modules on the same SafeKit cluster or when you deploy farm and mirror modules on two different clusters.

15.4 Dedicated replication network example

The attribute `ident="flow"` on the heartbeat, allows to identify the replication flow. For details, see 13.6 page 251.

conf/userconfig.xml - see 13 page 235

```
…
  <heart>
    <heartbeat name=”default” />
    <!– 2nd heartbeat special for dedicated replicated network -->
    <heartbeat name=”repli” ident="flow" />
  </heart>
…
```

15.5 Network load balancing examples in a farm module

15.5.1 TCP load balancing example

With the following **`userconfig.xml`** configuration file, you are defining a farm of 3 servers with network load balancing and failover on TCP services 9010 (SafeKit web service), 23 (Telnet), 80 (HTTP), 443 (HTTPS), 8080 (HTTP proxy) and 389 (LDAP).

With HTTP and HTTPS, network load balancing is set on the client IP address (`"on_addr"`) and not on the client TCP port (`"on_port"`), to ensure that the same client is always on the same server over several TCP connections (stateful versus stateless servers: see 1.4 page 19)

`conf/userconfig.xml` - see 13 page 235

```
<!DOCTYPE safe>
<safe>
<service mode="farm">
  <farm>
    <lan name=”net3” />
  </farm>
  <vip>
    <interface_list>
      <interface check="on" arpreroute=”on>
        <virtual_interface type="vmac_directed">
          <virtual_addr addr="192.168.1.50" where="alias" />
        </virtual_interface>
      </interface>
    </interface_list>
    <loadbalancing_list>
       <group name="tcpservices" >
         <cluster>
           <host name="node1" power="1" />
           <host name="node2" power="1" />
           <host name="node3" power="1" />
         </cluster>
         <rule port="9010" proto="tcp" filter="on_port" />
         <rule port="23"   proto="tcp" filter="on_port" />
         <rule port="80"   proto="tcp" filter="on_addr" />
         <rule port="443"  proto="tcp" filter="on_addr" />
         <rule port="8080" proto="tcp" filter="on_addr" />
         <rule port="389"  proto="tcp" filter="on_port" />
       </group>
    </loadbalancing_list>
  </vip>
</service>
</safe>
```

15.5.2 UDP load balancing example

With the following **userconfig.xml** configuration file, you are defining a farm of 3 servers with network load balancing and failover on UDP services 53 (DNS), 1645 (RADIUS).

conf/userconfig.xml – see 13 page 235

```
<!DOCTYPE safe>
<safe>
<service mode="farm">
  <farm>
    <lan name=”net3” />
  </farm>
  <vip>
    <interface_list>
      <interface check="on">
        <virtual_interface type="vmac_invisible">
          <virtual_addr addr="192.168.1.50" where="alias" />
        </virtual_interface>
      </interface>
    </interface_list>
    <loadbalancing_list>
       <group name="udpservices" >
         <cluster>
           <host name="node1" power="1" />
           <host name="node2" power="1" />
           <host name="node3" power="1" />
         </cluster>
         <rule port="53"    proto="udp" filter="on_ipid" />
         <rule port="1645"  proto="udp" filter="on_ipid" />
       </group>
    </loadbalancing_list>
  </vip>
</service>
</safe>
```

Note

With "on_ipid", the load balancing is made on the IP identifier filed in the packet IP header. The load balancing works even if the client always presents the same client IP address and client port at input.

15.5.3 Multi-group load balancing example

With the following **userconfig.xml** configuration file, you are defining a farm of 3 servers with a priority for HTTP traffic on the 1st server, HTTPS on the 2nd server and proxy HTTP on the 3rd server.

conf/userconfig.xml - see 13 page 235

```
<!DOCTYPE safe>
<safe>
<service mode="farm">
  <farm>
    <lan name=”net3” />
  </farm>
  <vip>
    <interface_list>
      <interface check="on" arpreroute=”on”>
        <virtual_interface type="vmac_directed">
          <virtual_addr addr="192.168.1.50" where="alias" />
        </virtual_interface>
      </interface>
    </interface_list>
    <loadbalancing_list>
       <group name="http_service" >
         <cluster>
           <host name="node1" power="3" />
           <host name="node2" power="1" />
           <host name="node3" power="1" />
         </cluster>
         <rule port="80"   proto="tcp" filter="on_addr" />
       </group>
       <group name="https_service" >
         <cluster>
           <host name="node1" power="1" />
           <host name="node2" power="3" />
           <host name="node3" power="1" />
         </cluster>
         <rule port="443"   proto="tcp" filter="on_addr" />
       </group>
       <group name="httpproxy_service" >
         <cluster>
           <host name="node1" power="1" />
           <host name="node2" power="1" />
           <host name="node3" power="3" />
         </cluster>
         <rule port="8080"   proto="tcp" filter="on_addr" />
       </group>
    </loadbalancing_list>
  </vip>
</service>
</safe>
```

15.6 Virtual hostname example with `vhost.safe`

The demonstration module `vhost.safe` shows how to set a virtual hostname (for details, see 13.8 page 269)

`conf/userconfig.xml` - see 13 page 235

```
…
  <vhost>
    <virtualhostname name="virtualname" envfile="vhostenv.cmd" />
  </vhost>
…
```

In addition to this configuration, special commands must be executed in the user scripts. Below is an example of Windows scripts. For Linux, please refer to the `vhost.safe` delivered with the Linux package.

`bin/start_prim.cmd` – see 14 page 293

```
@echo off
rem Script called on the primary server for starting application services
rem For logging into SafeKit log use:
rem "%SAFE%\safekit" printi | printe "message"
rem stdout goes into Application log
echo "Running start_prim %*"

rem Set virtual hostname
CALL "%SAFEUSERBIN%\vhostenv.cmd"

rem Next commands use the virtual hostname
FOR /F %%x IN ('hostname') DO SET servername=%%x
echo "hostname is "%servername%

rem WARNING: previous virtual hostname setting is insufficient to change the
hostname for services
rem If one service needs the virtual hostname, you need also to uncomment the rem
following

rem "%SAFE%\private\bin\vhostservice" SERVICE_TO_BE_DEFINED

set res=0

rem Fill with your services start call

set res=%errorlevel%

if %res% == 0 goto end

:stop
"%SAFE%\safekit" printe "start_prim failed"
rem uncomment to stop SafeKit when critical
rem "%SAFE%\safekit" stop -i "start_prim"

:end
```

bin/stop_prim.cmd - see 14 page 293

```
@echo off
rem Script called on the primary server for stopping application services
rem For logging into SafeKit log use:
rem "%SAFE%\safekit" printi | printe "message"
rem ----------------------------------------------------------
rem
rem 2 stop modes:
rem
rem - graceful stop
rem   call standard application stop with net stop
rem
rem - force stop (%1=force)
rem   kill application's processes
rem
rem ----------------------------------------------------------
rem stdout goes into Application log
echo "Running stop_prim %*"

set res=0

rem Reset virtual hostname
CALL "%SAFEUSERBIN%\vhostenv.cmd"

rem Next commands use the real hostname
FOR /F %%x IN ('hostname') DO SET servername=%%x
echo "hostname is "%servername%
```

```
rem default: no action on forcestop
if "%1" == "force" goto end

rem Fill with your services stop call
rem If necessary, uncomment to wait for the stop of the services
rem "%SAFEBIN%\sleep" 10

if %res% == 0 goto end

"%SAFE%\safekit" printi "stop_prim failed"

:end
rem WARNING: if the virtual hostname was set for services in start_prim.cmd,
rem uncomment the following to restore the real hostname in last stop phase :

rem "%SAFE%\private\bin\vhostservice" SERVICE_TO_BE_DEFINED
```

15.7 Software error detection example with `softerrd.safe`

The `softerrd.safe` module is a demonstration of the software error detection for mirror architecture (for configuration details , see 13.9 page 271)**.**

The module monitors the presence of:

- `mybin` and `myappli` started/stopped on the primary node with `start_prim`/`stop_prim`
- `myotherbin` started/stopped on the secondary node with `start_second`/`stop_second`

Detecting the shutdown of:

- `mybin` causes the module to `restart`
- `myappli` causes the execution of a special handler `restart_myappli.cmd`. This script increments the `maxloop` counter and restarts the `myappli` process
- `myotherbin` causes a stop of the module

The tests consist in killing the `mybin`, `myotherbin` or `myappli` processes with the `safekit kill` command.

Below is an extract of `softerrd.safe` for Windows. For Linux, look at the one delivered with the Linux package.

`conf/userconfig.xml` - see 13 page 235

```
…
      <errd>
         <proc name="mybin.exe" atleast="1" action="restart" class="prim"/>
         <proc name="myotherbin.exe" atleast="1" action="stop" class="second"/>
         <proc name="myappli.exe" atleast="1" action="restart_myappli"
class="myappli"/>
      </errd>
…
```

`bin/start_prim.cmd` - see 14 page 293

Note the call to `%SAFE%\safekit errd enable myappli` for starting the monitoring of the processes with `class="myappli"`

```
@echo off

%SAFE%\safekit printi "start mybin"
start %SAFEUSERBIN%\mybin.exe 10000000

%SAFE%\safekit printi "start myappli"
start %SAFEUSERBIN%\myappli.exe 10000000
%SAFE%\safekit errd enable myappli

:end
```

`bin/stop_prim.cmd` - see 14 page 293

Note the call to `%SAFE%\safekit errd disable myappli` for stopping the monitoring of the processes with `class="myappli"`

```
@echo on

rem default: no action on forcestop
if "%1" == "force" goto end

%SAFE%\safekit printi "stop mybin"
%SAFE%\safekit kill -level="terminate" -name="mybin.exe"

%SAFE%\safekit printi "stop myappli"
%SAFE%\safekit errd disable myappli
%SAFE%\safekit kill -level="terminate" -name="restart_myappli.cmd"
%SAFE%\safekit kill -level="terminate" -name="myappli.exe"

:end
```

`bin/restart_myappli.cmd`

Note the increment of the loop counter and the stop of the module when `maxloop` is reached

```
@echo off

rem Template for script called by errd on error detection instead of standard
restart
%SAFE%\safekit printi "restart_myappli"

rem first disable monitoring of the application
%SAFE%\safekit errd disable myappli

rem increment loop counter
%SAFE%\safekit incloop -i "restart_myappli"
if  %errorlevel% == 0 goto next
rem max loop reached
%SAFE%\safekit stop -i "restart_myappli"
%SAFEBIN%\exitcode 0

:next
rem max loop not reached : go on restarting the application
%SAFE%\safekit printi "Restart myappli"
```

```
%SAFE%\safekit kill -level="terminate" -name="myappli.exe"
start %SAFEUSERBIN%\myappli.exe 10000000

rem finally, enable monitoring of the application
%SAFE%\safekit errd enable myappli
```

15.8 TCP checker example

Below is an example of tcp checker definition that tests the Apache web service (for configuration details, see 13.11 page 278).

The default action when the tcp service is down is to restart locally the module (see 13.18.5 page 290 for the default failover rules description).

conf/userconfig.xml - see 13 page 235

```
  …
  <check>
    <tcp
         ident="Apache_80"
         when="both"
    >
         <to
         addr="172.21.10.5"
         port="80"
         interval="120"
         timeout="5"
         />
    </tcp>
   </check>
…
```

15.9 Ping checker example

The next example is the configuration of a ping checker that tests a router at 192.168.1.1 IP address (for configuration details, see 13.12 page 280). The default action when the router is down is to stop locally the module and to wait for the ping to be up (see 13.18.5 page 290 for the default failover rules description).

conf/userconfig.xml - see 13 page 235

```
…
<check >
  <ping ident="router">
  <to addr="192.168.1.1"/>
  </ping>
</check>
…
```

15.10 Interface checker example

Below is the example of an interface checker configuration automatically generated when `<interface check="on">` is set (for configuration details, see 13.5 page 243). In the `userconfig.xml`, the virtual IP address is defined as follows:

conf/userconfig.xml - see 13 page 235

```
<vip>
 <interface_list>
  <interface check="on">
   <real_interface>
    <virtual_addr addr="192.168.1.32" where="one_side_alias"/>
   </real_interface>
  </interface>
 </interface_list>
</vip>
```

The default action when the interface checker is down is to stop locally the module and to wait for the interface to be up (see 13.18.5 page 290 for the default failover rules).

To generate the configuration of the interface checker, SafeKit computes the hardware network interface, network and first IP address corresponding to the virtual IP address.

⇨ configuration generated in Windows

```
<check>
 <intf when="pre" ident="192.168.1.0"
       intf="{8358A0EE-2F3F-4FEE-A33B-EDC406C0C858}">
  <to local_addr="192.168.1.228"/>
 </intf>
</check>
```

Where `{8358A0EE-2F3F-4FEE-A33B-EDC406C0C858}` is the identity of the network interface for the network 192.168.1.0 and with the IP address 192.168.1.228 as first IP address (`safekit –r vip_if_ctrl –L`).

⇨ configuration generated in Linux

For instance, a configuration generated on Linux is:

```
<check>
 <intf when="pre" ident="192.168.1.0" intf="eth2">
  <to local_addr="192.168.1.20"/>
 </intf>
</check>
```

where eth2 is the identity of the network interface for the network 192.168.1.0 with the IP address 192.168.1.20 as first IP address (all this information is get from the `ifconfig –a` `ipconfig` or `ip addr show` command).

For configuration details, see 13.13 page 281.

15.11 IP checker example

Below is the example of an ip checker configuration automatically generated when `<virtual_addr check="on" …>` is set (for configuration details, see 13.5 page 243). In the `userconfig.xml`, the virtual IP address is defined as follows:

`conf/userconfig.xml` - see 13 page 235

```
…
<vip>
 <interface_list>
  <interface check="on" arpreroute="on">
   <real_interface>
    <virtual_addr addr="192.168.1.99" where="one_side_alias" check="on"/>
   </real_interface>
  </interface>
 </interface_list>
</vip>
…
```

The default action when the ip checker is down is to stopstart locally the module (see 13.18.5 page 290 for the default failover rules).

➪ configuration generated in Windows and Linux

The ip checker configuration generated is (for more information, see 13.14 page 282):

```
<check>
 <ip ident="192.168.1.99" when="prim">
  <to addr="192.168.1.99"/>
 </ip>
</check>
```

15.12 Custom checker example with `customchecker.safe`

The `customchecker.safe` module is a demonstration module of a custom checker (see 13.15 page 283).

- ➪ This custom checker tests the presence of a file on the primary server (`when="prim"`). The associated resource is called `custom.checkfile` (`ident="checkfile"`). It is set to `up` (file present) or `down` (file missing)
- ➪ The associated failover rule (configured in `<failover>`), is named `custom_failure` and causes the module to `restart` if the resource is `down` (see 13.18.5 page 290 for failover rules)

This example can be used as a basis for writing your own checker.

`conf/userconfig.xml` - see 13 page 235

```
…
      <check>
         <custom ident="checkfile" exec="checker.ps1"
                 arg="c:\safekit\checkfile" when="prim"/>
      </check>
      <user>
      </user>
      <failover>
         <![CDATA[
         custom_failure:
         if( custom.checkfile == down ) then restart();
         ]]>
      </failover>
…
```

`bin/checker.ps1`

Note the call to `safekit set -r custom.checkfile -m AM` to set the resource status (`up` or `down`)

```
param([Parameter(Mandatory = $true, ValueFromPipeLine = $true,
position=1)][String]$ModName,
      [Parameter(Mandatory = $true, ValueFromPipeLine = $true,
position=2)][String]$RName,
      [Parameter(Mandatory = $true, ValueFromPipeLine = $true,
position=3)][String]$Arg1Value,
      [Parameter(Mandatory = $false, ValueFromPipeLine = $false,
position=4)][String]$Grace="2",
      [Parameter(Mandatory = $false, ValueFromPipeLine = $false,
position=5)][String] $Period="5"
      )
# return up on success | down on failure
Function test([String]$Arg1Value)
{
        $res="down"
    # Replace the following by your test
    if (Test-Path "$Arg1Value")
        {
        $res="up"
        }
        return $res
}

$customchecker=$MyInvocation.MyCommand.Name
$safekit="$env:SAFE/safekit.exe"
$safebin="$env:SAFEBIN"
$gracecount=0
$prevrstate="unknown"
# wait a little
Start-Sleep $Period

while ($true){
        Start-Sleep $Period
        $rstate = test($Arg1Value)
        if($rstate -eq "down"){
                $gracecount+=1
        }else{
                $gracecount = 0
                if($prevrstate -ne $rstate){
                        & $safekit set -r "$RName" -v $rstate -i
$customchecker -m $ModName
                        $prevrstate = $rstate
                }
        }
        if($gracecount -ge $Grace){
                if($prevrstate -ne $rstate){
                        & $safekit set -r "$RName" -v $rstate -i
$customchecker -m $ModName
                        $prevrstate = $rstate
                }
                $gracecount = 0
        }
}
```

The executable associated with the checker is automatically called with at least 2 arguments:

- The 1st argument is the module name
- The 2nd is the name of the resource to be assigned

If the `<custom>` configuration contains the `arg` attribute, its value is passed as the next arguments.

The checker script is written with the following precautions:

- The resource is only assigned if its value has changed
- When the resource is down, the checker consolidates this state (`grace` times) before assigning it. This can help to avoid false error detections.

Important

Each time you modify the custom checker script in `SAFE/modules/AM/bin/`, you must apply the new configuration.

15.13 Module checker example with `leader.safe` and `follower.safe`

This example describes the two application modules `leader.safe` and `follower.safe` delivered with SafeKit:

- The leader module defines shared SafeKit resources between followers like virtual IP addresses and replicated directories
- The follower modules contain individual start and stop of several applications that are then isolated in different modules. Each follower module can be started and stopped independently without stopping the other modules.

The leader module is configured for a mirror architecture. It also includes the start and stop of the follower modules.

Each follower module is configured for a light architecture with user scripts and error detectors. The follower modules depend on the leader failover with the following module checker:

`follower/conf/userconfig.xml` - see 13 page 235

```
<check>
  <module name="leader"/>
</check>
```

This is a shortcut for:

```
<module name="leader">
 <to addr="127.0.0.1" port="9010"/>
</module>
```

Important

If you change the listening port for the SafeKit web service (as described in 10.6 page 167), replace the short configuration with the full one and change the port value.

16. SafeKit cluster in the cloud

You can install, configure, and administer SafeKit modules that run on virtual servers in the cloud instead of on-premises physical servers. This requires a minimum of cloud and/or server settings, especially to implement the virtual IP address. These settings are automatically done with SafeKit AWS CloudFormation template and Azure templates. Templates provide a very fast and easy way to install and preconfigure a SafeKit cluster in AWS, Azure or Google clouds.

For a quick start, refer to:

✓ mirror cluster in AWS or farm cluster in AWS

✓ mirror cluster in Azure or farm cluster in Azure

✓ mirror cluster in GCP or farm cluster in GCP

16.1 SafeKit cluster in Amazon AWS

In the following, we suppose that you are familiar with:

⇨ Amazon Elastic Compute Cloud (Amazon EC2) that offers computing capacity in the Amazon Web Services (AWS) cloud. For more information about the features of Amazon EC2, see the Amazon EC2 product page

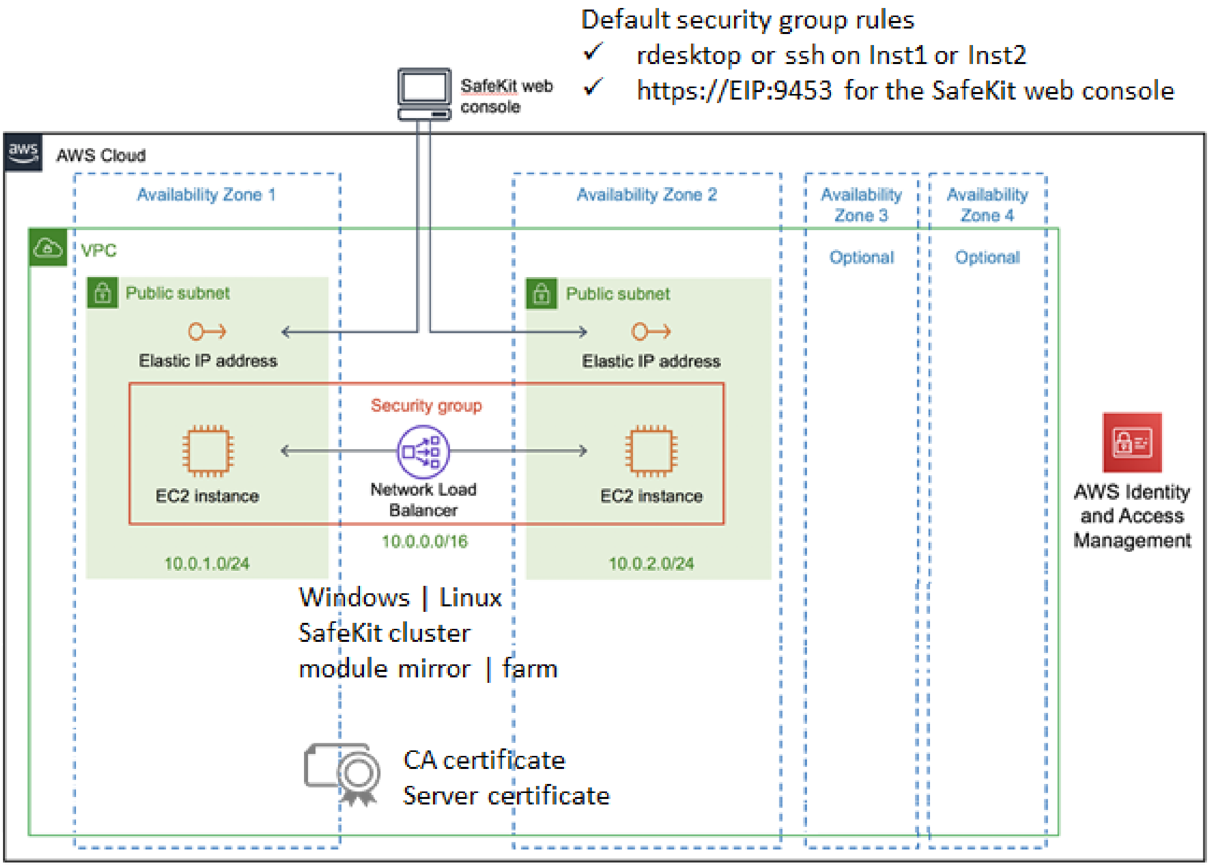
⇨ AWS CloudFormation that helps deploying instances and applications on Amazon EC2. It permits to save a lot of time and effort so that you can spend less time managing EC2 resources and more time focusing on your applications that run in AWS.

16.1.1 Install a SafeKit cluster with the AWS CloudFormation template for SafeKit

SafeKit provides an AWS CloudFormation templates for AWS QuickStart that are a very fast and easy way for implementing the SafeKit clustering solution. It offers 2 templates:

⇨ one template for deploying a mirror cluster, with some specific settings described in 16.1.3 page 318

⇨ one template for deploying a farm cluster, with some specific settings described in 16.1.4 page 319



- it deploys two EC2 instances (up to four) in the same region but distributed across multiple availability zones. You can choose the instance type and operating system (Windows 2016 or CentOS 7)
- it configures one virtual network (virtual private cloud, VPC)
  - ✓ it includes a public address (Elastic IP, EIP) and a private address for each instance. The private addresses are used for the SafeKit framework communications
  - ✓ It configures the AWS security groups for accepting remote connection from the administrator (remote desktop for Windows, ssh for Linux) and from the SafeKit web console (https://EIP:9453). It also accepts all communications on the private addresses.
- It configures an AWS load balancer for implementing the virtual IP of the mirror or the farm module according to the chosen template
- it runs all operations to make SafeKit ready for use:
  - ✓ it installs the SafeKit package
  - ✓ it fills the SafeKit cluster configuration and applies it on all nodes (for details on cluster configuration, see 12 page 227)
  - ✓ it applies the HTTPS configuration for securing the SafeKit web console (for details on HTTPS configuration, see 11 page 177)

- ✓ it installs, configures, and starts a mirror or farm module according to the chosen template

At the end of the SafeKit AWS CloudFormation template deployment, simply connect a web browser to the URL specified in the model's application output. This URL connects to the configuration wizard described in 11.4.3 page 197. Apply the described procedure for using the secure SafeKit web console with your browser (connected to https://EIP:9453, where EIP is the public address of one cluster node). You can then exploit SafeKit as an on-premises installation by installing, configuring, and administering a mirror or farm module in the AWS cloud.

16.1.2 Install a SafeKit cluster without the AWS CloudFormation template for SafeKit

You can implement SafeKit on AWS instances created outside the AWS CloudFormation template for SafeKit. In this case, before implementing a SafeKit module, the administrator must manually make settings for AWS, instances, and SafeKit. Then you have specific settings for implementing your SafeKit module:

- ➪ for mirror cluster, see 16.1.3 page 318
- ➪ for farm cluster, see 16.1.4 page 319

AWS settings

You must set AWS to:

- ➪ associate public addresses to each instance if you want to administer them with the SafeKit web console from the internet
- ➪ configure the security groups associated with network(s) to enable the communications of the SafeKit framework and the SafeKit web console. The ports to open are described in 10.3.3.2 page 159
- ➪ use a high-bandwidth, low-latency network if real-time replication is used in a mirror module

Instances settings

In each instance, you must also:

- ➪ install the SafeKit package
- ➪ apply the HTTPS configuration to secure the SafeKit web console (described in 11 page 177)

SafeKit settings

Finally, you must enter the SafeKit cluster configuration and apply it to all nodes (for details on cluster configuration, see 12 page 227). For each network, it can be specified if it can be used by the console and/or the framework. By default, a network can be used by both the console and the framework (`console = "on" framework = "on"`). In the case of the public network accessible from the internet, it is preferable not to use it for the communications of the SafeKit framework but only for the console (`console = "on" framework = "off"`).

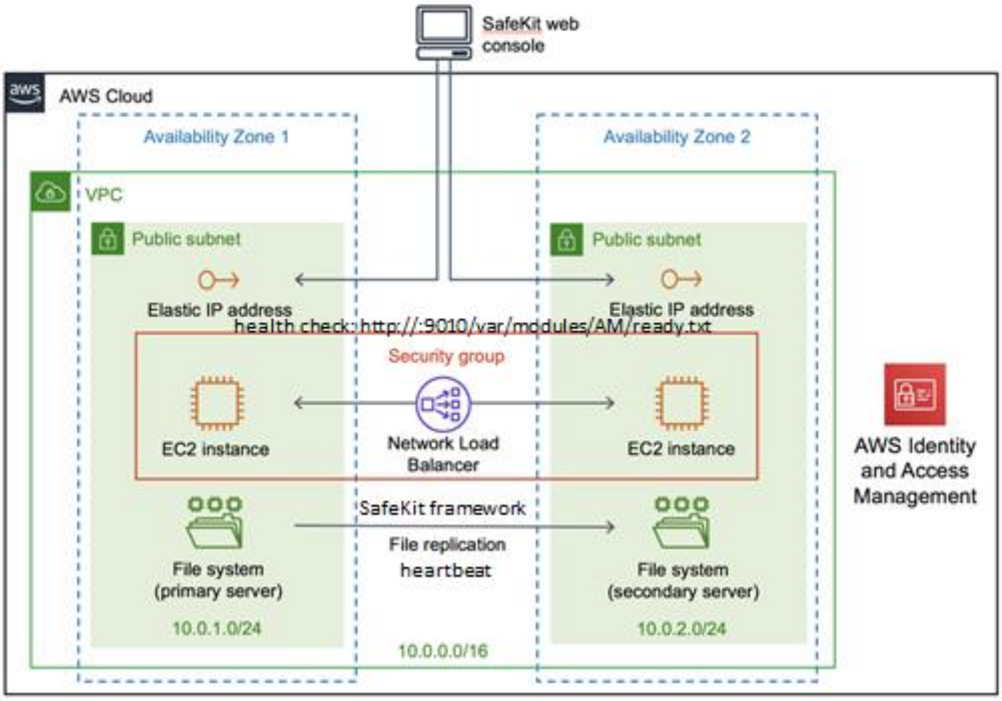
For example, the SafeKit cluster configuration file would be:

```
<cluster>
<lans>
<lan name="Public" console="on" framework="off">
<node name="Server1" addr="18.214.97.59"/>
<node name="Server2" addr="52.5.205.73"/>
</lan>
<lan name="Private" console="on" framework="on">
<node name="Server1" addr="10.0.1.10"/>
<node name="Server2" addr="10.0.2.10"/>
</lan>
</lans>
</cluster>
```

The first `lan` definition is only for the SafeKit web console; the second one is also for the SafeKit framework between cluster nodes.

16.1.3 Mirror cluster in AWS

Mirror module features are operational in the AWS cloud (real-time file replication, failover, process death detection, checkers, …), except the virtual IP address failover. Anyway, you can set up a SafeKit mirror module on the cluster and use the Elastic load balancing provided by AWS (see Elastic load balancing products in AWS) in such way that all the traffic is routed only to the primary node. An IP address and/or DNS name is associated with the load balancer that plays the role of the virtual IP. The AWS CloudFormation template for SafeKit configures a network load balancer and applies all the required setup. You just must set the load balancing rule and security group for your application.



If you set up the mirror module outside the AWS CloudFormation template for SafeKit, you must configure yourself the AWS load balancer and the security group.

For the load balancer, you must:

- specify the rules for your application
- set the SafeKit cluster nodes in the target group
- configure the `health check`. This one tests whether the instance is in a healthy state or an unhealthy state.

The load-balancer routes the traffic only to healthy instances. It resumes routing requests to the instance when this one has been restored to a healthy state.

SafeKit provides a health checker for SafeKit modules. For this, configure it in the load balancer with:

- HTTP protocol
- port 9010, the SafeKit web service port
- URL `/var/modules/AM/ready.txt`, where AM is the module name

In a mirror module, the health checker:

- returns `OK`, that means that the instance is healthy, when the module state is ✓ `PRIM` (green) or ✓ `ALONE` (green)
- returns `NOT FOUND`, that means that the instance is out of service, in all other states

The AWS network security group must be at least configured to enable communications for the following protocols and ports:

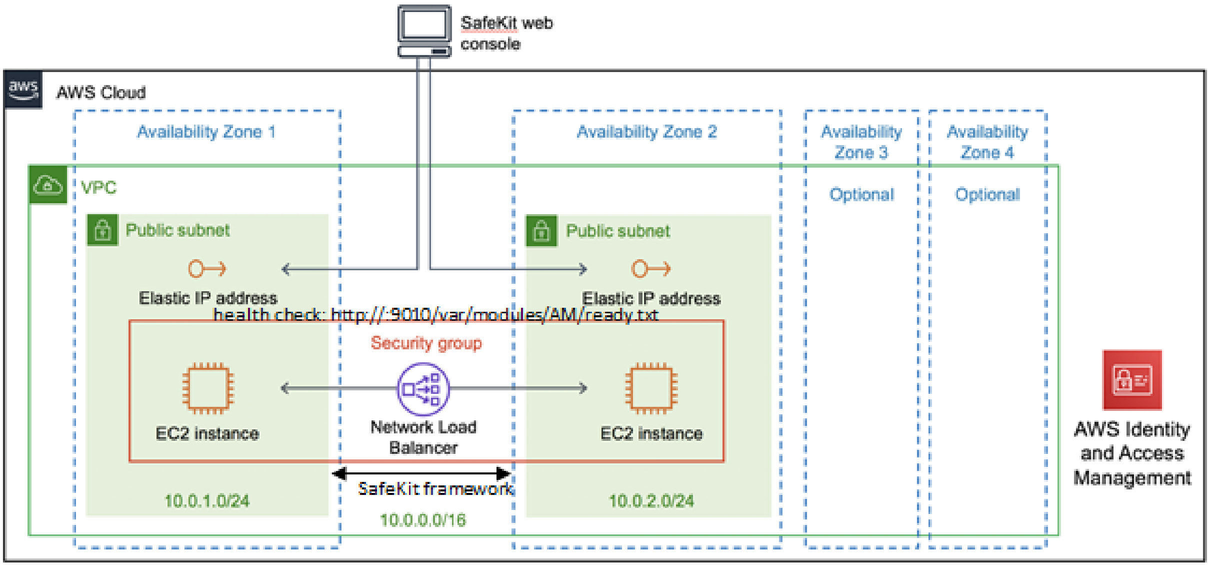
- UDP - 4800 for the `safeadmin` service (between SafeKit cluster nodes)
- UDP - 8888 for the module heartbeat (between SafeKit cluster nodes)
- TCP – 5600 for the module real time file replication (between SafeKit nodes)
- TCP – 9010 for the load-balancer health check and the SafeKit web console in HTTP
- TCP – 9453 for the SafeKit web console in HTTPS
- TCP – 9001 for configuring the SafeKit web console for HTTPS

Important

The module's port value depends on the module id (for details, see 10.3.3.2 page 159).The previous values are the one for the first module installed on the node.

16.1.4 Farm cluster in AWS

Most farm module features are operational in the AWS cloud (process death detection, checkers), except the virtual IP address with load-balancing. Anyway, you can set up a SafeKit farm module on the cluster and use the Elastic load balancing provided by AWS (see Elastic load balancing products in AWS). An IP address and/or DNS name is associated with the load balancer that plays the role of the virtual IP. The AWS CloudFormation template for SafeKit configures a network load balancer and applies all the required setup. You just must set the load balancing rule and security group for your application.



If you set up the farm module outside the AWS CloudFormation template for SafeKit, you must configure yourself the AWS load balancer and the security group.

For the load balancer, you must:

- specify the rules for your application
- set the SafeKit cluster nodes in the target group
- configure the `health check`. This one tests whether the instance is in a healthy state or an unhealthy state.

The load-balancer routes the traffic only to healthy instances. It resumes routing requests to the instance when this one has been restored to a healthy state.

SafeKit provides a health check for SafeKit modules. For this, configure it in the load balancer with:

- HTTP protocol
- port 9010, the SafeKit web service port
- URL `/var/modules/AM/ready.txt`, where AM is the module name

In a farm module, the health check:

- returns `OK`, that means that the instance is healthy, when the module state is ✓ `UP` (green)
- returns `NOT FOUND`, that means that the instance is out of service, in all other states

The AWS network security group must be at least configured to enable communications for the following protocols and ports:

- UDP - 4800 for the `safeadmin` service (between SafeKit cluster nodes)
- TCP – 9010 for the load-balancer health check and the SafeKit web console in HTTP
- TCP – 9453 for the SafeKit web console in HTTPS
- TCP – 9001 for configuring the SafeKit web console for HTTPS

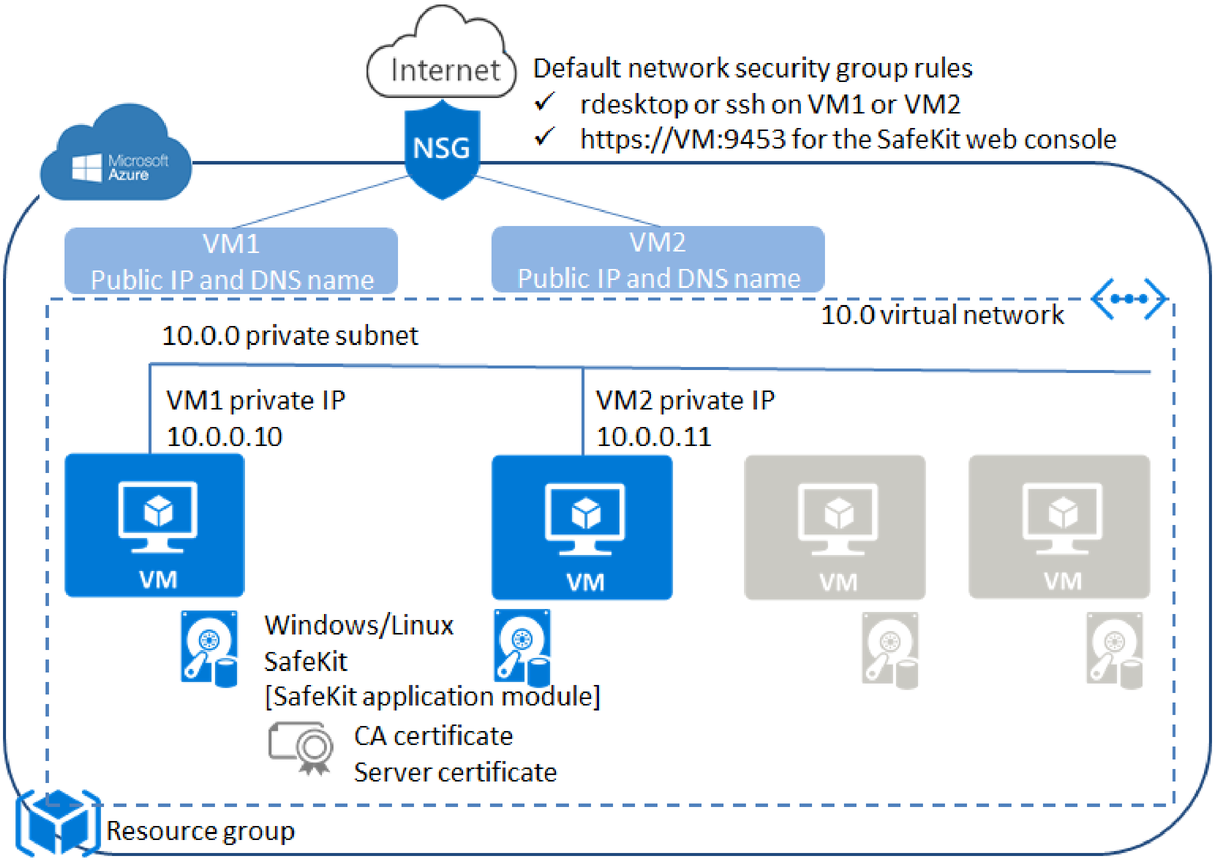
16.2 SafeKit cluster in Microsoft Azure

In the following, we suppose that you are familiar with Microsoft Azure that is a cloud computing service created by Microsoft for building, testing, deploying, and managing applications and services through a global network of Microsoft-managed data centers. For more information about the features and use of Azure, see the Microsoft Azure portal.

16.2.1 Install a SafeKit cluster with the Azure resource template for SafeKit

SafeKit provides an Azure resource template that is a very fast and easy way for implementing the SafeKit clustering solution. It offers 2 templates:

- one template for deploying a mirror cluster, with some specific settings described in 16.2.3 page 323
- one template for deploying a farm cluster, with some specific settings described in 16.2.4 page 325



- it deploys a resource group that defines all the resources necessary for implementing the SafeKit cluster in one location but separate availability zones
- the resource group contains two virtual machines (up to four) running in different availability zones. You can choose the operating system (Windows 2016, Linux CentOS 7). Each virtual machine has an internet access through a public IP address and DNS name
- the resource group contains a private virtual network for SafeKit framework communications

- it configures the network security group for accepting only remote connection from the administrator (remote desktop for Windows, ssh for Linux) and from the SafeKit web console (https://DNS:9453) on the public addresses.
- It configures an Azure load balancer for implementing the virtual IP of the mirror or the farm module according to the chosen template
- it runs all operations to make SafeKit ready for use:
  - ✓ it installs the SafeKit package
  - ✓ it fills the SafeKit cluster configuration and applies it on all nodes (for details on cluster configuration, see 12 page 227)
  - ✓ it applies the HTTPS configuration for securing the SafeKit web console (for details on HTTPS configuration, see 11 page 177)
  - ✓ it installs, configures, and starts a mirror or farm module according to the chosen template

At the end of the SafeKit Azure template deployment, simply connect a web browser to the URL specified in the template’s application output. This URL connects to the configuration wizard described in 11.4.3 page 197. Apply the described procedure for using the secure SafeKit web console with your browser (connected to https://DNS:9453, where DNS is the DNS name of one cluster node). You can then exploit SafeKit as an on-premises installation by installing, configuring, and administering a mirror or farm module in the Azure cloud.

16.2.2 Install a SafeKit cluster without the Azure resource template for SafeKit

You can implement SafeKit on Azure virtual machines created outside the Azure resource template for SafeKit. In this case, before implementing a SafeKit module, the administrator must manually make settings for Azure, virtual machines and SafeKit. Then you have specific settings for implementing your SafeKit module:

- for mirror cluster, see 16.2.3 page 323
- for farm cluster, see 16.2.4 page 325

Azure settings

You must set Azure to:

- associate public IP addresses and DNS name to virtual machines if you want to administer them with the SafeKit web console from the internet
- configure the network security group to enable the communications of the SafeKit framework and the SafeKit web console. The ports to open are described in 10.3.3.2 page 159
- use a high-bandwidth, low-latency network if real-time replication is used in a mirror module

Virtual machines settings

On each virtual machine, you must also:

- install the SafeKit package

- apply the HTTPS configuration to secure the SafeKit web console (described in 11 page 177)

SafeKit settings

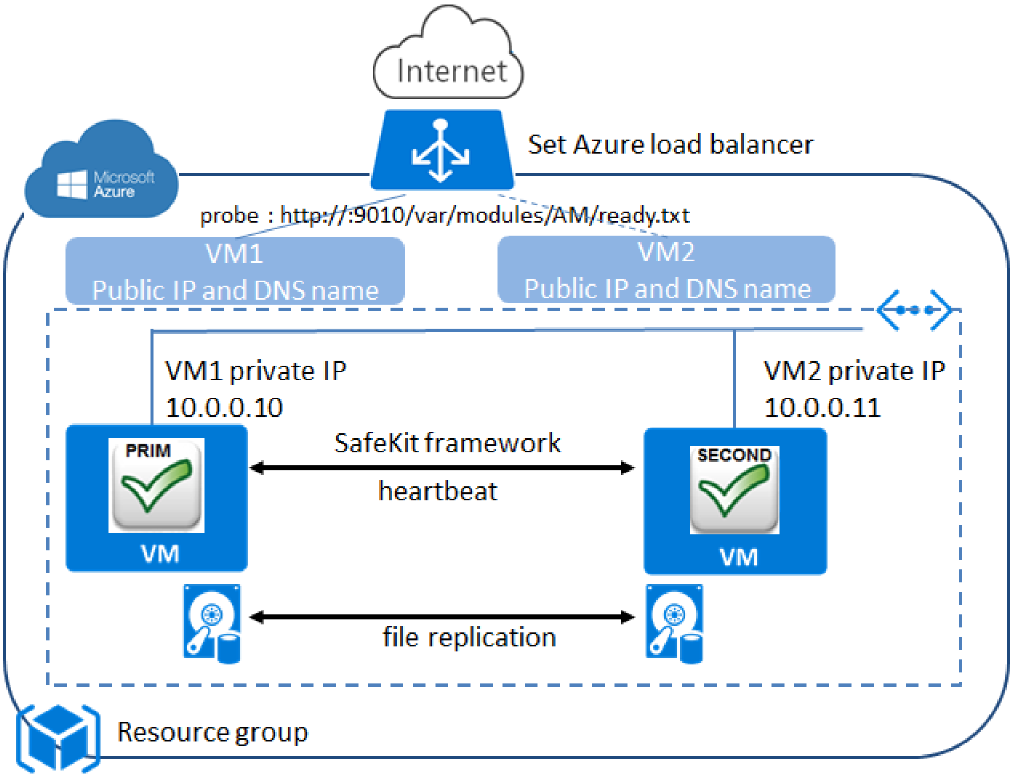
Finally, you must enter the SafeKit cluster configuration and apply it to all nodes (for details on cluster configuration, see 12 page 227). For each network, it can be specified if it can be used by the console and/or the framework. By default, a network can be used by both the console and the framework (`console = "on" framework = "on"`). In the case of the public network accessible from the internet, it is preferable not to use it for the communications of the SafeKit framework but only for the console (`console = "on" framework = "off"`). For example, the SafeKit cluster configuration file would be:

```
<cluster>
<lans>
<lan name="Public" console="on" framework="off">
<node name="Server1" addr="centosazurlinvm1.westeurope.cloudapp.azure.com"/>
<node name="Server2" addr="centosazurlinvm2.westeurope.cloudapp.azure.com"/>
</lan>
<lan name="Private" console="on" framework="on">
<node name="Server1" addr="10.0.0.10"/>
<node name="Server2" addr="10.0.0.11"/>
</lan>
</lans>
</cluster>
```

The first `lan` definition is only for the SafeKit web console; the second one is also for the SafeKit framework between cluster nodes.

16.2.3 Mirror cluster in Azure

Mirror module features are operational in the Azure cloud (real-time file replication, failover, process death detection, checkers, …) except the virtual IP address failover. Anyway, you can set up a SafeKit mirror module on the cluster and use the load balancing provided by Azure (see Load Balancer in Azure) and route request only to the primary node. An IP is associated with the load balancer that plays the role of the virtual IP. The Azure resource template for SafeKit configures a network load balancer and applies all the required setup. You just must set the load balancing rule and network security group for your application.



If you set up the mirror module outside the Azure resource template for SafeKit, you must configure yourself the Azure load balancer and the network security group.

For the load balancer, you must:

- ➪ specify the rules for your application
- ➪ set the SafeKit cluster nodes into the backend pool
- ➪ configure the `probe`. This one tests whether the instance is in a healthy state or an unhealthy state.

The load balancer routes traffic only to healthy instances. It resumes routing requests to the instance when the instance has been restored to a healthy state.

SafeKit provides a probe for SafeKit modules. For this, configure the probe in the load balancer with:

- ➪ HTTP protocol
- ➪ port 9010, the SafeKit web service port
- ➪ URL `/var/modules/AM/ready.txt`, where AM is the module name

In a mirror module, the probe:

- ➪ returns `OK`, that means that the instance is healthy, when the module state is ✓ `PRIM` (green) or ✓ `ALONE` (green)
- ➪ returns `NOT FOUND`, that means that the instance is out of service, in all other states

The Azure network security group must be at least configured to enable communications for the following protocols and ports:

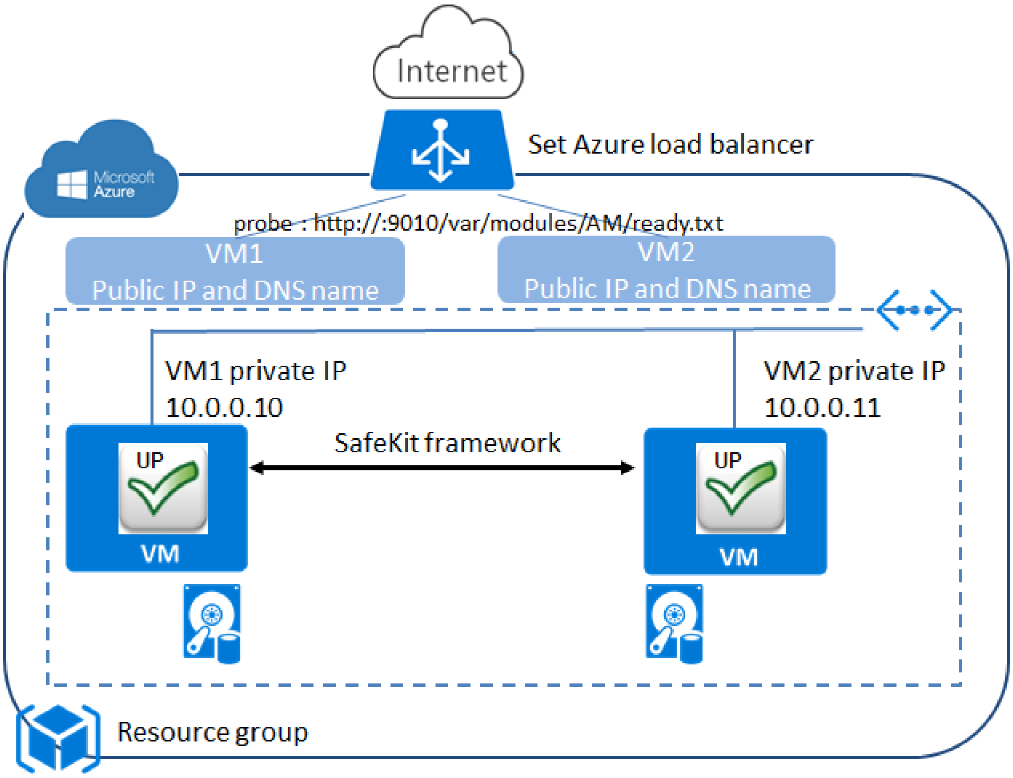
- UDP - 4800 for the `safeadmin` service (between SafeKit cluster nodes)
- UDP - 8888 for the module heartbeat (between SafeKit cluster nodes)
- TCP – 5600 for the module real time file replication (between SafeKit nodes)
- TCP – 9010 for the load-balancer health check and the SafeKit web console in HTTP
- TCP – 9453 for the SafeKit web console in HTTPS
- TCP – 9001 for configuring the SafeKit web console for HTTPS

Important

The module's port value depends on the module id (see 10.3.3.2 page 159).The previous values are the one for the first module installed on the node.

16.2.4 Farm cluster in Azure

Most farm module features are operational in the Azure cloud (process death detection, checkers), except the virtual IP address with load-balancing. Anyway, you can set up a SafeKit farm module on the cluster and use the load balancing provided by Azure (see Load Balancer in Azure). An IP is associated with the load balancer that plays the role of the virtual IP. The Azure resource template for SafeKit configures a network load balancer and applies all the required setup. You just must set the load balancing rule and network security group for your application.



If you set up the farm module outside the Azure resource template for SafeKit, you must configure yourself the Azure load balancer and the network security group.

For the load balancer, you must:

- specify the rules for your application
- set the SafeKit cluster nodes as backend
- configure the `probe`. This one tests whether the instance is in a healthy state or an unhealthy state.

The load balancer routes traffic only to healthy instances. It resumes routing requests to the instance when the instance has been restored to a healthy state.

SafeKit provides a probe for SafeKit modules. For this, configure the probe in the load balancer with:

- HTTP protocol
- port 9010, the SafeKit web service port
- URL `/var/modules/AM/ready.txt`, where AM is the module name

In a farm module, the probe:

- returns `OK`, that means that the instance is healthy, when the farm module state is ✓ `UP` (green)
- returns `NOT FOUND`, that means that the instance is out of service, in all other states

The Azure network security group must be at least configured to enable communications for the following protocols and ports:

- UDP - 4800 for the `safeadmin` service (between SafeKit cluster nodes)
- TCP – 9010 for the load-balancer health check and the SafeKit web console in HTTP
- TCP – 9453 for the SafeKit web console in HTTPS
- TCP – 9001 for configuring the SafeKit web console for HTTPS

16.3 SafeKit cluster in Google GCP

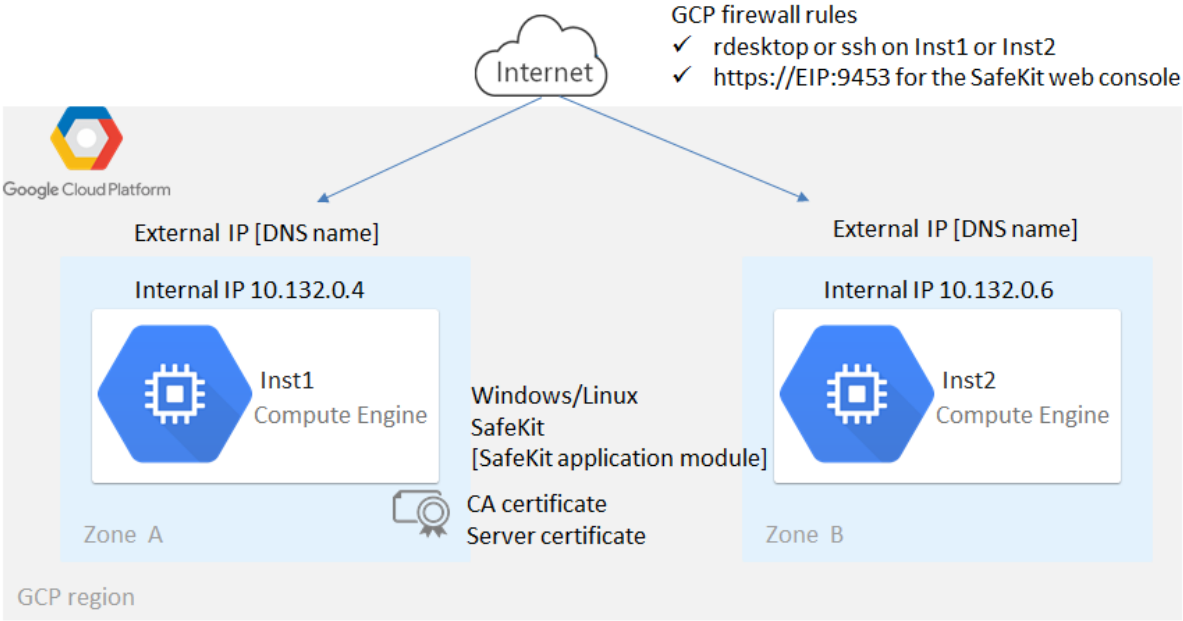
In the following, we suppose that you are familiar with Google Cloud Platform (GCP) that delivers virtual machines running in Google's innovative data centers and worldwide fiber network. For more information about the features and use of Google Cloud Platform, see the Google Cloud Computing documentation.

16.3.1 Install a SafeKit cluster with the Google Marketplace solution for SafeKit

SafeKit provides solutions in the Google Marketplace that are a very fast and easy way for implementing the SafeKit clustering solution. It offers 4 solutions:

- ➩ 2 solutions for deploying a mirror cluster (one solution for Windows and one solution for Linux), with some specific settings described in 16.3.3 page 329
- ➩ 2 solutions for deploying a farm cluster (one solution for Windows and one solution for Linux), with some specific settings described in 16.3.4 page 331

See Startup Guide for a full description of these solutions' deployment.



- ➩ it deploys two virtual machine instances in the same region but distributed across two zones. You can choose the instance type and operating system (Windows 2019 or CentOS 7)
- ➩ it uses one virtual network (virtual private cloud, VPC) attached to the project in which the solution is deployed
  - ✓ it includes a public address (External IP, EIP) and a private address for each instance. The private addresses are used for the SafeKit framework communications
  - ✓ It configures the firewall for accepting remote connection from the administrator (remote desktop for Windows, ssh for Linux) and from the SafeKit web console (https://EIP:9453). It also accepts all communications on the private addresses.

- It configures an GCP load balancer for implementing the virtual IP of the mirror or the farm module according the chosen solution
- it runs all operations to make SafeKit ready for use:
  - ✓ it includes the SafeKit package into the VM image
  - ✓ it fills the SafeKit cluster configuration and applies it on all nodes (for details on cluster configuration, see 12 page 227)
  - ✓ if HTTPS is selected for the deployment, it applies the HTTPS configuration for securing the SafeKit web console (for details on HTTPS configuration, see 11 page 177)
  - ✓ it installs, configures, and starts a mirror or farm module according to the chosen solution

At the end of the SafeKit Google Marketplace solution deployment, simply follow the recommendations listed into "Suggested next steps". This must be done if you have selected HTTPS for the SafeKit console access mode. You must connect a web browser to the URL specified for opening the configuration wizard described in 11.4.3 page 197. Apply the procedure for using the secure SafeKit web console with your browser (connected to https://EIP:9453, where EIP is the public address of one cluster node). You can then exploit SafeKit as an on-premises installation by installing, configuring, and administering a mirror or farm module in the Google GCP cloud.

16.3.2 Install a SafeKit cluster without the Google Marketplace solution for SafeKit

You can implement SafeKit on Google virtual machines created outside the Google Marketplace solution for SafeKit. In this case, before implementing a SafeKit module, the administrator must manually make settings for Google Compute Engine, virtual machines and SafeKit. Then you have specific settings for implementing your SafeKit module:

- for mirror cluster, see 16.3.3 page 329
- for farm cluster, see 16.3.4 page 331

GCP settings

You must set GCP to:

- associate an external IP address (and optionally DNS name) to each virtual machine instance if you want to administer them with the SafeKit web console from the internet
- configure the firewall rules for the Virtual Private Cloud (VPC) network to enable the communications of the SafeKit framework and the SafeKit web console. The ports to open are described in 10.3.3.2 page 159
- use a high-bandwidth, low-latency network if real-time replication is used in a mirror module

Virtual machines settings

On each virtual machine, you must also:

- install the SafeKit package

- apply the HTTPS configuration to secure the SafeKit web console (described in 11 page 177)

SafeKit settings

Finally, you must enter the SafeKit cluster configuration and apply it to all nodes (for details on cluster configuration, see 12 page 227). For each network, it can be specified if it can be used by the console and/or the framework. By default, a network can be used by both the console and the framework (`console = "on" framework = "on"`). In the case of the public network accessible from the internet, it is preferable not to use it for the communications of the SafeKit framework but only for the console (`console = "on" framework = "off"`).

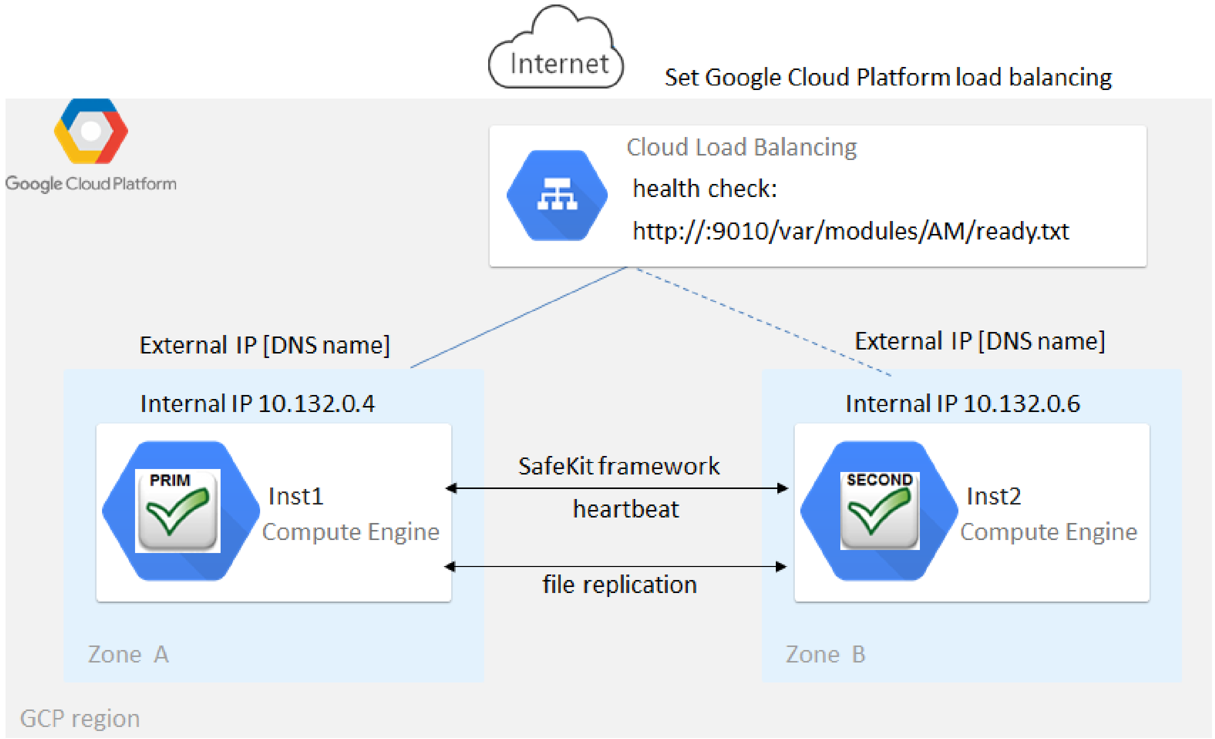
For example, the SafeKit cluster configuration file would be:

```
<cluster>
<lans>
<lan name="Public" console="on" framework="off">
<node name="Inst1" addr="104.199.111.158"/>
<node name=" Inst2" addr="35.205.22.195"/>
</lan>
<lan name="Private" console="on" framework="on">
<node name=" Inst1" addr="10.132.0.4"/>
<node name=" Inst2" addr="10.32.0.6"/>
</lan>
</lans>
</cluster>
```

The first `lan` definition is only for the SafeKit web console; the second one is also for the SafeKit framework between cluster nodes.

16.3.3 Mirror cluster in GCP

Mirror module features are operational in the Google Cloud Platform (real-time file replication, failover, process death detection, checkers, …) except the virtual IP address failover. Anyway, you can set up a SafeKit mirror module on the cluster and use the load balancing provided by GCP (see Load Balancer in GCP) and route request only to the primary node. An IP is associated with the load balancer that plays the role of the virtual IP. The Google Marketplace solution for SafeKit configures a network load balancer and applies all the required setup. You just have to set the load balancing rule and network security group for your application.



If you set up the mirror module outside the Google Marketplace solution for SafeKit, you must configure yourself the Google load balancer and the network firewall.

For the load balancer, you must:

- specify the rules for your application
- set the SafeKit cluster nodes as backend
- configure the `health check`. This one tests whether the instance is in a healthy state or an unhealthy state.

The load balancer routes traffic only to healthy instances. It resumes routing requests to the instance when the instance has been restored to a healthy state.

SafeKit provides a health check for SafeKit modules. For this, configure the health check in the load balancer with:

- HTTP protocol
- port 9010, the SafeKit web service port
- URL `/var/modules/AM/ready.txt`, where AM is the module name

In a mirror module, the health check:

- returns `OK`, that means that the instance is healthy, when the module state is ✓ `PRIM` (green) or ✓ `ALONE` (green)
- returns `NOT FOUND`, that means that the instance is unhealthy, in all other states

The network firewall must be at least configured to enable communications for the following protocols and ports:

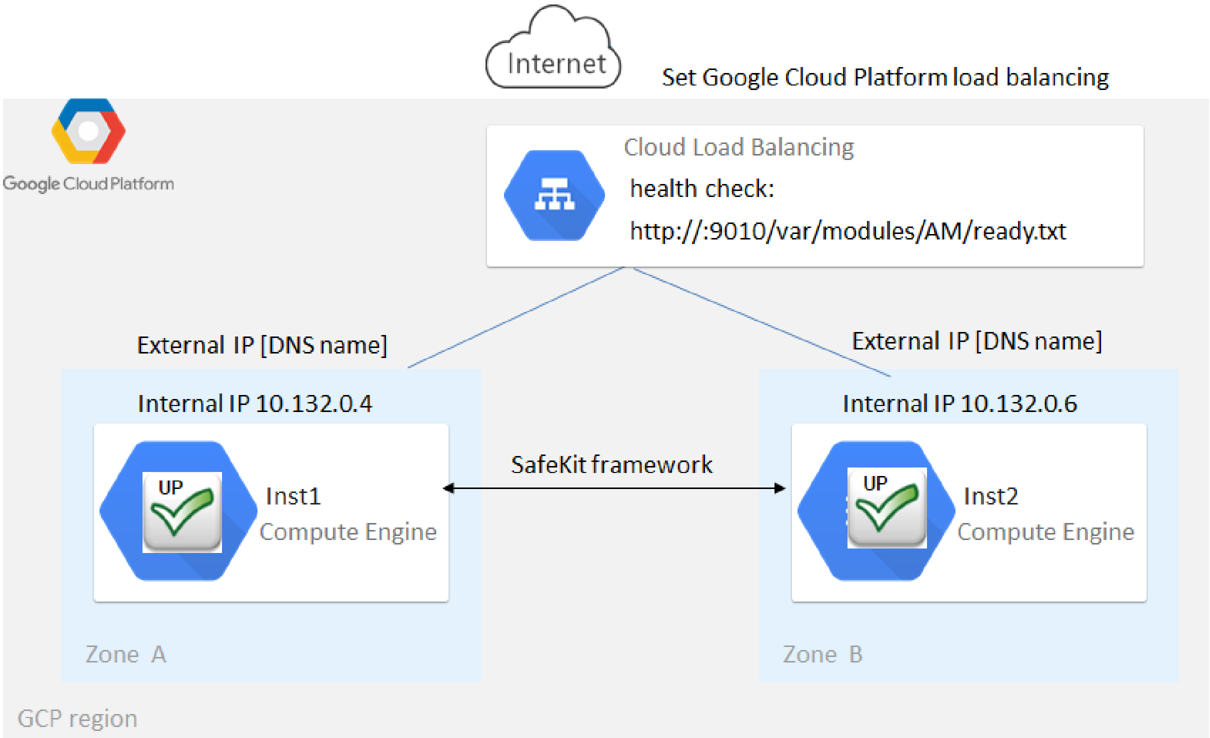
- UDP - 4800 for the `safeadmin` service (between SafeKit cluster nodes)
- UDP - 8888 for the module heartbeat (between SafeKit cluster nodes)
- TCP – 5600 for the module real time file replication (between SafeKit nodes)
- TCP – 9010 for the load-balancer health check and the SafeKit web console in HTTP
- TCP – 9453 for the SafeKit web console in HTTPS
- TCP – 9001 for configuring the SafeKit web console for HTTPS

Important

The module's port value depends on the module id (see 10.3.3.2 page 159).The previous values are the one for the first module installed on the node.

16.3.4 Farm cluster in GCP

Most farm module features are operational in the Google Cloud Platform (process death detection, checkers), except the virtual IP address with load-balancing. Anyway, you can set up a SafeKit farm module on the cluster and use the load balancing provided by GCP (see Load Balancer in GCP). An IP is associated with the load balancer that plays the role of the virtual IP. The Google Marketplace solution for SafeKit configures a network load balancer and applies all the required setup. You just have to set the load balancing rule and network security group for your application



If you set up the farm module outside the Google Marketplace solution for SafeKit, you must configure yourself the Google load balancer and the network firewall.

For the load balancer, you must:

- specify the rules for your application

- set the SafeKit cluster nodes as backend
- configure the `health check`. This one tests whether the instance is in a healthy state or an unhealthy state.

The load balancer routes traffic only to healthy instances. It resumes routing requests to the instance when the instance has been restored to a healthy state.

SafeKit provides a health check for SafeKit modules. For this, configure the health check in the load balancer with:

- HTTP protocol
- port 9010, the SafeKit web service port
- URL `/var/modules/AM/ready.txt`, where AM is the module name

In a farm module, the health check:

- returns `OK`, that means that the instance is healthy, when the farm module state is ✓ `UP` (green)
- returns `NOT FOUND`, that means that the instance is out of service, in all other states

The network firewall must be at least configured to enable communications for the following protocols and ports:

- UDP - 4800 for the `safeadmin` service (between SafeKit cluster nodes)
- TCP – 9010 for the load-balancer health check and the SafeKit web console in HTTP
- TCP – 9453 for the SafeKit web console in HTTPS
- TCP – 9001 for configuring the SafeKit web console for HTTPS

17. Third-Party Software

SafeKit uses the third-party software listed below. For licenses details, refer to the links or the license files into the `SAFE/licenses` directory (`SAFE=/opt/safekit` in Linux and `SAFE=C:\safekit` in Windows if `%SYSTEMDRIVE%`=C:).

libxml	http://xmlsoft.org MIT license - http://www.xmlsoft.org/FAQ.html#License Used by the SafeKit framework
libxslt	http://xmlsoft.org/XSLT/ MIT license - https://gitlab.gnome.org/GNOME/libxslt/blob/master/Copyright Used by the SafeKit framework
Net-SNMP	http://net-snmp.sourceforge.net BSD like and BSD license - http://www.net-snmp.org/about/license.html Used by SafeKit SNMP agent
HTTP server	https://httpd.apache.org/ Apache license - https://www.apache.org/licenses/LICENSE-2.0 Used by the SafeKit web service for the web console, the deprecated java console, the distributed commands, and the checkers between modules
APR	https://apr.apache.org/ Apache license - https://www.apache.org/licenses/LICENSE-2.0 Used by the Apache HTTP server
PCRE	http://www.pcre.org/ BSD license - https://www.pcre.org/licence.txt Used by the Apache HTTP server
libexpat	https://github.com/libexpat/libexpat BSD license - https://github.com/libexpat/libexpat/blob/master/expat/COPYING Used by the Apache HTTP server
cURL	http://curl.haxx.se Curl license - https://github.com/curl/curl/blob/master/docs/LICENSE-MIXING.md Used by the distributed commands and the module checker
cgic	ANSI C library for CGI Programming Cgic license - Credits and License Terms Used by the SafeKit web service
OpenSSL	http://www.openssl.org

	dual OpenSSL and SSLeay license - https://www.openssl.org/source/license.html Used when securing the web console, the distributed commands, and the checkers between modules
Lua	http://www.lua.org MIT license - https://www.lua.org/license.html Used by `safekit config` command and the web console
Info-ZIP	http://info-zip.org BSD like license - http://infozip.sourceforge.net/license.html Used to pack/unpack a .safe template
libnet	Packet Construction and Injection Libnet license - license Used for arpreroute and ping

SafeKit uses the following third-party packages only for the SafeKit web console:

jquery	http://jquery.org/ MIT license - https://jquery.org/license/ jQuery is a fast, small, and feature-rich JavaScript library
jquery-ui	http://jqueryui.com/ MIT license - https://github.com/jquery/jquery-ui/blob/master/LICENSE.txt jQuery UI is a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library
jquery-lang	https://github.com/Irrelon/jquery-lang-js MIT license - https://github.com/Irrelon/jquery-lang-js/blob/master/js/jquery-lang.js Used for translating the messages of the web console from a default language to other languages
jquery-ui-tabs.paging	jquery-ui-tabs-paging - MIT license Used for translating the messages of the web console from a default language to other languages
codemirror	http://codemirror.net/ MIT-style license - https://github.com/codemirror/CodeMirror/blob/master/LICENSE Text editor widget by Marijn Haverbeke. Used in the SafeKit Web console for file edition
codemirror-ui	https://github.com/jagthedrummer/codemirror-ui

	MIT License - https://github.com/jagthedrummer/codemirror-ui/blob/master/LICENSE Simple interface written by Jeremy Green to act as a wrapper around the codemirror widget
bootstrap icons	https://icons.getbootstrap.com/ - MIT license – https://github.com/twbs/bootstrap/blob/v4.0.0/LICENSE

Thanks to iTweek (http://itweek.deviantart.com/) for the Knob buttons toolbar icons.

Log Messages Index

“Action …” messages

File replication and reintegration messages

Index

Architectures

Installation

Console

Advanced Configuration

Administration

Support

Other