Nokia Siemens Networks
Security

Certificate Authority for Network Elements

Platform Architecture Description

Version 2.1

© 2008 Nokia Siemens Networks. All rights reserved.

Table of Content

© 2010 Nokia Siemens Networks. All rights reserved.

© 2010 Nokia Siemens Networks. All rights reserved.

1. Introduction

1.1 Executive Summary

Nokia Siemens Networks provides a cost-effective and easy-to-use Public Key Infrastructure (PKI) solution for issuing and managing X.509v3 digital certificates. The lifecycle management of issued certificates can be automated and adapted to operator procedures.

The described Public Key Infrastructure is based on the INSTA DefSec Oy Certifier® software product. INSTA DefSec is part of the INSTA group (http://www.insta.fi/en) and has signed official and exclusive partnership with NSN, for joint development of a TelCo environment capable PKI.

In order to run the INSTA software in a TelCo operator environment, a highly availability, scalable architecture is defined by NSN in agreement with INSTA. This architecture defines following possible deployments:

- A simple single-Server setup to use for basic testing, see Paragraph 2.1.
- A simple HA setup (2BE) with minimum traffic separation, see Paragraph 2.2
- A security-scaled HA setup using multi-layered traffic separation, see paragraph 2.3.

1.2 Scope of this document

Following Chapters describe and define the NSN INSTA PKI HAPF 2.0 Platform version. It is defined by NSN and used along with INSTA Certifier 5.x releases. A new version of the HAPF will be defined and maintained by NSN for later INSTA Certifier releases.

A description and definition goes as far as to name all technical details needed to build the platform out of predefined components (such as, for example, INSTA rpms or standard RedHat packages) and to describe the configuration of the OS during an automated setup procedure. A Description of the architecture of the Certifier application as well as instructions on taking CA Software into use is not in the scope of this document.

© 2010 Nokia Siemens Networks. All rights reserved.

2. PKI Deployment Scenarios

In TelCo environments, PKI is required to be highly available and mature servers/operating systems with long lifetime and little maintenance efforts are needed. Other factors like small racking/floorspace footprint or low energy consumption are highly appreciated.

NSN has defined standard “HW layouts” with predefined OS settings on which NSN INSTA PKI is to be operated. Those are described in the following. For each layout, INSTA certifier is running on RedHat Linux 6.2 ES.

2.1 Single Server Setup

Single Server Setup is designed to use the same hardware as it is delivered for the regular HA TelCo Platform. But instead of using a full-blown server Hardware, Single Server can also run on any laptop or other computer, even on vmware player ®.

This setup is ideal when starting trials or in an early deployment phase. If the same supported Hardware is used as for the commercial production systems, then Single-Server can be smoothly extended towards a full-blown HA deployment. This means, changes when going from Single Server to HA Platform *can* be transparent to EEs.

On the other hand, if a later deployment in production would anyway start from scratch or if only demonstration purposes are followed, then any available COTS hardware that is able to run x86_64 versions of RedHat Enterprise Server Linux can be used for PKI demos on the authentic OS setup. Differences between HA deployments and demo-setups are normally in the security and scalability of perimeter filtering, rather than in the performance.

2.1.1 Setup characteristics

Following mainly characterizes Sinlge-Server setup:

- If continuation of the Single-Server into a full-scale HA is planed, then an appropriate hardware platform and sufficient disk-space should be chosen from the beginning. A migration procedure ensures continuation of the key- and certificate/CRL usage also after Hardware upgrades.
- If continuation of the Single-Server into a full-scale HA is planed, this must be considered for IP addressing and hostname assignment! The Single-Server should be used as what will later become “BE1” role and use CA-trunk VLAN IP addresses that will later be used by the frontends for the corresponding services in question. (see also chapter 5.1). Otherwise, a migration can not be supported and re-installation will be needed.
- If later reuse of the CAs and issued certificates from the Single-Server is planned for the full-blown HA deployment, a Hardware Security Module (HSM) is highly recommended to ensure proper control over the CA keys. This fact significantly reduces the applicability of e.g. virtualized or non-standard server deployments!

© 2010 Nokia Siemens Networks. All rights reserved.

- Single-Server with default hardware as recommended and purchased via NSN satisfies minimum HA requirements such as redundant power supplies, RAID controllers, hot-plugable disks and partly scalable networking. It is also fully supported for the migration into a full-HA deployment.
- Minimum traffic separation is approached when having OAM connectivity on a dedicated interface and using an 802.1q connection for the different PKI management and enrollment or publishing procedures. This requires that the server provides at least 3 LAN interfaces. However, Single Server will also work and be supported with only one Ethernet interface. In this case, no traffic separation at all is happening.
- If later expansion to HA setups is planned, four Ethernet interfaces will be needed! The Addresses and VLANs are reused after a possible migration.
- Performance of the single-server solution can be expected to be almost as high as in the full setup. Restrictions might apply only in networking throughput if single-interface setups are used, or if non-default hardware is used. Nevertheless, it lacks HA for Certifier services and the security full traffic separation. Only therefore it is not recommended for TelCo production environments.

2.2 Simple HA setup

Simple HA setup can be used either for extended trials which last longer in time or will certainly be upgraded into production. Or for operators who do not assign high priority to secure PKI operations. Simple HA comprises 2 fully functional PKI servers (Backends, BE) in an Active/Standby cluster. In order to be fully functional, the PKI needs a signing engine, communication stacks to receive/answer signing requests and a database to store CA keys, issued Certificates and CRL. If the Insta Software is configured to run in a Backend (BE) role, servers provide all the listed functions and act as full PKI.

Compared to the Single Server setup, Simple HA provides full redundancy and can continue in case an entire server fails. Compared to Full-HA, the Single Server misses a scalable security approach since it communicates directly with end entities.

2.2.1 Setup Characteristics

Following mainly characterizes Simple-HA setup:

- A fully functional, fully performing and highly available PKI system is provided. However: It does not comply to best-practice security, because a direct connection from the (yet untrusted) end entities to the machine holding the keys will be needed.
- Continuation or upgrading of the Simple HA into a full-scale HA is even simpler than in the Single-Server case with default Hardware! Because only an additional 2 or 4 servers will be installed to serve as “Bastion Hosts” or “Frontends (FE)” to the existing installation. However, IP addressing and

© 2010 Nokia Siemens Networks. All rights reserved.

hostname assignment should have been planned in a way, that they only need to be extended (i.e. new addresses or VLANs can be added), but not changed or components being removed.

- Especially in cases where Simple HA is used to serve production networks, a Hardware Security Module (HSM) is most highly recommended! The fact that EEs can directly access the CA server creates an additional need for security “on the host” and HSM adds to this significantly. Without an HSM module, all CA private keys are stored in the BE Sybase database and thereby theoretically vulnerable to theft and decryption. HSM only allows PKCS#11 based access to the keys and requires 2nd factor (HSM PIN/passphrase) and a smartcard to be available, thereby significantly increasing the security level for key access!

- Full traffic separation is achieved on vertical level: OAM connectivity uses a dedicated interface and 802.1q connections allow different VLANs for the different PKI management and enrollment or publishing procedures. This is similar to the Single Server setup, with the addition, that in Single Server the 802.1q trunk is only an option. In Simple-HA, four LAN interfaces are needed and the trunk is no option but mandatory.

- Performance of the simple-HA solution can be expected to be as high as in the full setup. Nevertheless, it lacks full traffic separation on horizontal level and allows direct connection of untrusted entities to the CA server. Only therefore it is not recommended for TelCo production environments.

2.3 Full HA Setup

In the full HA setup, Insta separation of Frontend (FE) and Backend (BE) comes to use. Commonly, 4 or 6 servers are used and connected as redundant pairs. Inside each pair, failover services for the Certifier functionality and for the Certificate repository are provided. In addition, the Frontends act as bastion hosts communicating via an internal TLS-protected interface to the Backends.

Signing operations, key store and creation of CRL and certificate repository happens in the Backends. Communication stacks with untrusted entities are terminated in the Frontends and messages are relayed to the Backends. This way, additional filters and security barriers are brought in and create “horizontal traffic separation”

Standard recommended setup for Production Environments is the 4-Server full-HA deployment!

Other setups are either not supported (in case of Single-Server) or not recommended (as in case of the simple-HA) for production purposes!

© 2010 Nokia Siemens Networks. All rights reserved.

2.3.1 Setup Characteristics of standard deployment

Full HA setup in the recommended version with 1 FE-pair is characterized mainly by the following:

- One pair (BE) functions as database backend and Certifier engine. It is used to sign certificate requests and to store issued certificates. It also holds the private keys of the CAs or a connection to an HSM module where those are stored. BE HA pairs are active-standby HA with disk replication from the active to the standby node.

- A second pair (FE) functions as server frontend and runs the protocol stacks for the different certificate management protocols and enrollment requests. It communicates to the backend database via TLS (Server-side on the BE). Both FE servers are registered to the BE, but the connection is only active to one server that holds the active IP address. If any service or IP fails, the FE will switch over all services and IPs.

This separation of frontend into Backend allows complete separation of traffic from different origins or of different services towards the frontend ONLY, and further the relaying of requests from the frontend towards the backend ONLY. In other words, there is no way an EE would directly contact the backend database – unless you configure it explicitely in BE and FW – and thereby it is possible, to vertically and horizontally split traffic types and define perimeters to monitor and filter.

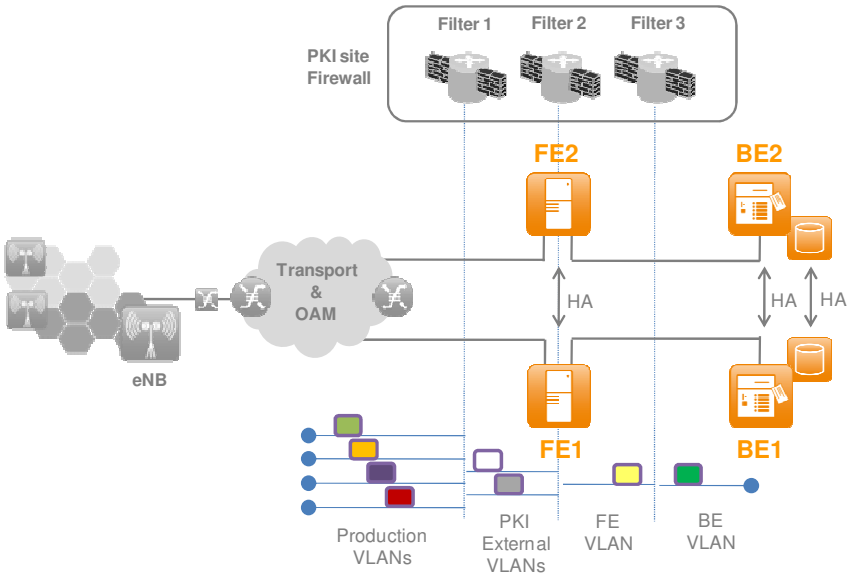


Figure 1: *High-level architecture and traffic separation of the PKI in full HA deployment*

© 2010 Nokia Siemens Networks. All rights reserved.

2.4 High Security Deployment

In some cases, it has been required to even separate the traffic on Frontends not only *"vertically"*, but also in a *"horizontal"* direction. Please refer to Chapter 5.3 to find an explanation of these terms and the principles of the networking solution of the PKI.

Scenario with an additional FE-pair are supported, but only useful in cases where the operator policy has extreme requirements towards traffic separation. See also the illustration in ***Figure 7:*** *Security Zones and VLAN "chain" from EEs towards Backend,* passing to understand better, how additional servers might help to improve security: Traffic from already known and thereby trusted EEs terminate on an "inside FE", while completely new and untrusted EEs will terminate on an "outside FE". That implicates, that e.g. CMP Initialization Requests or anonymous LDAP binds would be terminated on a more exposed "outside FE", while authenticated traffic from known entities that have already operator certificates (key-updates, CRL downloads etc …) will get through to a "inside FE" which has higher protection.

© 2010 Nokia Siemens Networks. All rights reserved.

3. PKI Hardware specification

In a full HA setup, the hardware and architecture must be setup as described in this document and with reference to the oracle documentation. While the detailed installation procedure and instructions are given in the installation guide, the specification of used/supported HW is made here.

3.1 Server Model Sun/Oracle Netra X4270

For reasons of pricing, lifetime and quality, NSN has decided to base and support the NSN INSTA PKI platform on Oracle – former Sun Netra X4270 servers. In the given configuration the servers have following characteristics:

- 2 x Xeon L5518 quad-Core processor 2,4 GHz 8MB-L3
- 2U rack height, 18 kg weight / 1540 BTU/h / 450W
- 18 DDR3 slots, comes with 16GB equipped (8x2GB)
- 4 builtin 1Gbps NICs plus ILOM
- comes as 2x300GB SAS-2 disk, DVD, RAID
- Solid & widely used TelCo platform, well established and field-proven

A complete description of the servers and datasheet can be downloaded from the oracle webpages:

http://www.oracle.com/us/products/servers-storage/servers/netra-carrier-grade/sun-netra-x4270-server-070754.html

3.2 Disk usage

3.2.1 RAID configuration

On all servers, underlying Harddisks are configured in a RAID 1 (mirroring) on HW level with the Netra's builtin controllers. This is transparent to the filesystem layout under Linux, where disks are accessed via common SCSI drivers as labeled partitions.

Instructions to set up RAID 1 are given in the Installation Guideline.

3.2.2 Disk partitioning

Disk Partitioning is applied during the installation of each server. While the actual partitions itself can't be modified (meaning: mountpoints and filesystem types are fix!), the actual size of each partition can be increased from its minimum towards whatever the disk space allows. For instructions to customize partition sizes during installation, please check the Installation Guide.

© 2010 Nokia Siemens Networks. All rights reserved.

On the BE servers, the following default partitioning is found after installation:

		minimum size [MB]	normal size [MB]
/dev/sda2	/	1.100	7.500
/dev/sda1	/boot	120	500
/dev/sda6	/backup	500	50.000
/dev/drbd1 (sda3)	/usr/local/certifier	1.500	150.000
/dev/sda5	[swap]	500	32.000

Figure 2: *Filesystems under Linux on the active PKI Backend (BE) server (passive has no DRBD)*

On the FE servers, the following default partitioning is found after installation:

		minimum size [MB]	normal size [MB]
/dev/sda2	/	1.100	7.500
/dev/sda1	/boot	120	500
/dev/sda6	/backup	500	50.000
/dev/sda3	/usr/local/certifsub	1.500	150.000
/dev/sda5	[swap]	500	32.000

Figure 3: *Filesystems under Linux on the active PKI Frontend (FE) server.*

All partitions are of type ext3, besides swap (is of type swap) with default size 32GB. Mind, that the numbering of the partitions can change, depending on the particular Linux patchlevel and the “relative” sizes.

On BE servers, the `/usr/local/certifier` filesystem is mounted by the HA solution under `/dev/drbd1`. On FE servers, there is no DRBD in use and `/usr/local/certifsub` is hence seen as regular filesystem on a normal partition.

The size of the `/backup` filesystem might vary, depending on the exactly delivered (or used) disksize. Because the linux setup procedure uses the “`–grow`” option to just assign all diskspace that is not used by other partitions to “`/backup`”

© 2010 Nokia Siemens Networks. All rights reserved.

3.3 Network Interfaces

Each physical Netra server is delivered with 4 RJ45/1Gbps LAN interfaces plus an ILOM port. Hence the 4-Server setup requiring 20 LAN ports in total, not counting inter-switch synchronization/trunking or uplinks. The use of the LAN interfaces is determined by the automated installation procedure and can't be changed! Cabling must be made accordingly:

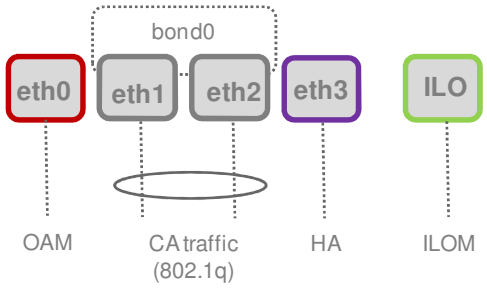


Figure 4: *Required LAN Interface assignment/cabling to traffic categories on Netra X4270 servers*

- ILOM traffic should be separated completely from any other traffic. It connects to the service processor board of the physical machines and can be either http(s) or ssh or both.
- HA traffic on eth3 might well be a direct link/crossover cable between the active and standby node. It comprises DRBD synchronization (on BE only) and also Linux heartbeat signals and failover (on BE and FE).
- OAM on eth0 is the first interface that is configured automatically during the installation. All later connections to the servers itself (i.e. root-shell) will be made via this interface. In order to prevent a single point-of-failure on the hosts, eth0 is used as a backup for eth3. However, there is no backup for OAM connectivity besides ILOM.
- Bonding on eth1 and eth2 is used to provide a connection for all other traffic that is not HA or OAM related. That is, all PKI services as certificate publishing, CMP etc… Principles of traffic separation inside this bond0 interface are explained in Chapter 5.3.

© 2010 Nokia Siemens Networks. All rights reserved.

4. Linux Installation Specification

NSN INSTA HAPF 2.0 uses RedHat Enterprise Linux 6.2 for x86_64 architecture. A basic license (for up to 2 sockets per machine) is sufficient. The Sizing and Ordering Guideline describes the process of how to order RedHat Linux licenses in NSN for customer projects. In practice, a medium will be delivered to the customer project which includes all needed SW – RedHat linux, INSTA SW and supporting configuration scripts to set up the solution – on a single USB Pendrive.

The customer has then to complete this media by adding a valid license file and customer specific planning parameters. This process is supported by MS-excel and VBA macros, see Installation Guidelines for details.

4.1 Installed Linux core components

When installing RedHat Linux, NSN uses so-called Anaconda kickstart files. They define a reproducible specification of the Linux installation and thus allow production of identical systems, differing only in a few site specific parameters like IP addresses, hostnames, timezones etc …

NSN INSTA HAPF2.0 Linux Core components to be installed are defined in the kickstart files. In order to get an identical package composite on RHEL 6.2 x86_64, the same `%packages` section is applied for all servers (BE and FE), see Figure 5.

Each line in the figure from "`aide`" to "`wget`" represents an rpm package from the used RedHat distribution. The "`--nobase`" option guarantees that no other rpms are installed, besides the absolute minimum required to run RedHat at all (for example and obviously, a kernel-package and glibc are needed as well and are installed installed, even if not mentioned in the `%packages` section).

Further, all packages that are "*dependencies*" to the listed ones must and will be installed as a prerequisite. For example, `net-snmp` requires the `net-snmp-libs` to be installed first. This will happen, even if `net-snmp-libs` is not in above list. A definition of requirements during building of the rpm files ensures that, and yum is taking care for it. After all, there will be approximately 265 packages installed on each PKI server, they are all listed in Appendix A along with version numbers.

```
%packages --nobase
aide
gnutls
libibverbs
librdmacm
libxslt
lsof
man
net-snmp
net-snmp-utils
nmap
```

© 2010 Nokia Siemens Networks. All rights reserved.

```
ntp
openhpi
openldap
openldap-servers
openldap-clients
openssh
openssh-clients
openssh-server
openssl
OpenIPMI-libs
perl-TimeDate
sos
strace
sysstat
tcpdump
vconfig
vlock
wget
yum
```

Figure 5: *Package section in the Anaconda kickstart file for NSN INSTA HAPF 2.0*

4.2 Additional Packages

After Anaconda Linux System Installer has completed the `-nobase` installation and added the core packages listed above, a set of additional packages is installed during the so-called `%post`-install phase. Those packages are installed with more or less “individual control” among revisions of the installation media.

For example, every time a new deliverable binary image to create install media is assembled, the latest kernel version and eventually other necessary patches can be added via this mechanism of “additional packages”. When new additional packages are added to the installation media in this way, the “MEDVERS” parameter in the stampfile.txt will show an increased version number.

Generally, this mechanism is not used to deliver patches to regular rpms, but in order to ensure kernel versions can be updated throughout deliverables and without depending on a change of the complete RHEL iso media. Further, the HA SW rpms like pacemaker or DRBD are brought to the system via this method – also to be independent of RedHat specific versions of this SW.

Patches that are available at the time of media freeze will also be added in a separate directory “rhpatches” and applied during installation.

© 2010 Nokia Siemens Networks. All rights reserved.

4.3 Basic OS settings at Installation

Most settings to the HA and OS configuration are defined in a planning parameter sheet and then automatically applied from the sheet into the OS configuration by automated install scripts. The procedure and background is described in the NSN INSTA PKI Installation Guideline.

Some settings are already predefined during the initial installation phase by Anaconda installer itself and not applied by later auto-configuration scripts. Those parameters are needed to do basic OS install and are listed below:

Parameter	change	Change in	Description
`timezone`	needed	Planning-sheet	Set UTC timezone
`lang en_US.UTF-8`	not supported	Kickstart file	System language
`keyboard us`	not supported	Kickstart file	Keyboard layout
`rootpw`	after install	Shell after login	Default password
`firewall –enabled`	optional	Kickstart or shell	Enable iptables
`selinux –disabled`	not supported	Kickstart or shell	Disables SELinux
`network eth0`	needed	Planning-sheet	OAM IP parameters
`part [*]`	optional	Planning-sheet	sizes of partitions

Figure 6: *Key system settings made during installation, change "Needed"can be done in the planning sheet, change "optional" can be done by editing the auto-generated kickstart files before installation (see install guide). Mind, that optional changes are not needed and system runs fine on default values (while it might not run fine on any non-default value). Change "not supported" means, that we have not tested it, will not test it in this release and will not evaluate errors that obviously result from changing it.*

4.4 Additional Users

During the Anaconda installation procedure and subsequent NSN rapid-setup phase, several users are created on the system, most of which are required by system daemons. For interactive login, the following logins are added:

```
nsn:x:501:501:Nokia Siemens Networks:/home/nsn:/bin/bash
certfier:x:502:2::/home/certfier:/bin/bash
```

Each of those logins has the password "*password*" by default. However, user `nsn` is mandatorily changing its password at the first login, for `root` and `certifier` this change is required to be executed by administrative users by using the "`passwd`" command.

© 2010 Nokia Siemens Networks. All rights reserved.

5. OS Configuration Details

The following chapter describes in each paragraph, how subsystems are configured by the auto-configuration scripts. The order how Auto-configuration scripts are called is briefly depicted in the Appendix C "Order of installation Actions". Before describing the parameter definitions made, the Auto-Configuration mechanism is described here for a better understanding.

5.1 Server Roles and connectivity

In the recommended 4-Server HA setup, 2 HA pairs – frontend (FE) and backend (BE) as described in 2.2 – comprise the PKI. Each server has a HA role and a PKI role:

PKI role	HA role	
FE1	act	*Frontend services for EE communication*
FE2	stby	*Take over resources of FE1 in failure case*
BE1	act	*Certifier engine and backend database*
BE2	stby	*Take over resources of BE1 in failure case*

Switchover scenarios are described in the HA Chapter 5.5. Each HA pair is hosted in a separate *"Security Zone"* and Perimeter Filters should be applied before the traffic reaches the FE's and between FE's and BE's. See the next paragraph for an explanation of how IP addressing and VLAN assignment should be done to support both, HA and secure traffic separation.

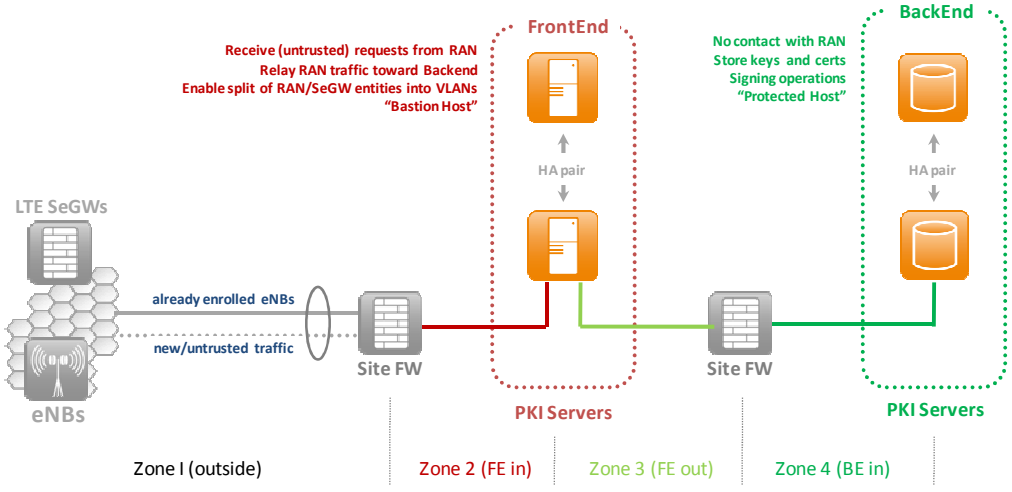


Figure 7: *Security Zones and VLAN "chain" from EEs towards Backend, passing 2 separate filters*

© 2010 Nokia Siemens Networks. All rights reserved.

5.2 Principle of Auto-Configuration

Required parameters such as IP addresses, passwords etc are taken from the Excel Planning Sheet and are transferred with a macro-tool into Operating System files. All planning parameters that have values assigned are thrown into a file called "`configure.sh`" which is created by the excel macro. In this file, the configuration parameters are shown as indexed names, like e.g. "`HOSTNAME[1]`" and are referred as array elements of Bash environment variables by the installation scripts as, e.g.:

```
${HOSTNAME[X]}
```

Where "`HOSTNAME`" is the parameter name (or: Array) as assigned in the excel sheets first column and the index `[X]` represents a particular HA role. In Bash, only numerical indexes are possible and hence the following table shows the mapping of "Roles" into "indexes".

ROLE	X	Y	Description
ss	0	-	Standalone Single Server (Testlab)
fe1	1	2	Default Active PKI Frontend Server
fe2	2	1	Default Standby PKI Frontend Server
be1	3	4	Default Active PKI Backend Engine
be2	4	3	Default Standby PKI Backend Engine
fe3	5	6	Additional Active PKI Frontend Server (high-secure deployment only)
fe4	6	5	Additional Standby PKI Frontend Server (high-secure deployment only)

Auto configuration scripts refer to the above Roles and Indexes of `configure.sh` when using the value that has been assigned to a parameter (i.e. to an array with that Index), for example:

```
${HOSTNAME[2]}="pki-fe2-host"
```

In configure.sh a shell-environment is set up for all installation routines. Scripts will be replacing every occurrence of `${HOSTNAME[X]}` with *pki-fe1-host* if `X=2` or with a correspondingly different hostname for other values of `X`. A script called "`setnsnenv.sh`" that is delivered with the installation media can be sourced in a regular shell to make the variables available for interactive working after installation.

Note: The Index "Y" is mentioned here only for completeness and used for convenience in the installation scripts. It refers to the (relative) HA-peer of each machine in a HA-pair. For example, the HA-peer of "be1" (X=3) is "be2" (X=4), and therefore, on be1 is Y=4 and on be2 is Y=3. This allows easy referring to the peer in a script.

© 2010 Nokia Siemens Networks. All rights reserved.

5.2.1 Meta-Environment settings

Some settings must be known to the installation procedure before it actually can start, as e.g. the location of the installation scripts or the number and order of scripts to run when setting up a particular role. Per fixed convention, those settings are defined in a file `/etc/hapf20.flag`. It might further be needed, to use functions of HAPF20 after the installation has ended, e.g. for updates or customer specific integrations. To provide this, a directory `/etc/hapf.d` is created and hosts all needed libraries and configuration files for further operation of HAPF2.0. The previously mentioned script setnsnenv.sh will set up the Meta-environment, then source all scripts defined in `/etc/hapf20.flag`

The Meta-Environment is defined by the macro `mkks61.exe` which is called by the planning excel sheet when it creates the kickstart and other config files like `syslinux.cfg` or `configure.sh`. It is then placed inside the kickstart files and simply transferred `into /etc/hapf20.flag` file by so-called “here-text” in the `%post` install phase. This way, the parameters that link together the different phases and components of an installation that happens in different servers become available to the setup procedure.

5.3 Networking Solution Principle

Because most customers prefer to use their existing routing/switching infrastructure in the server rooms rather than purchasing a new L2/L3 solution with every product, the NSN INSTA PKI is offered without a predefined L2/L3 hardware. If a connectivity solution for the PKI is explicitly requested by a customer, it is suggested, to use e.g. 2 SRX240 for both – L2 and L3 connectivity. A pure L2/routing solution without firewall functionality could be applied using e.g. Cisco Catalyst 3750. In that case, perimeter filtering must be applied in any other existing Firewall.

Ports and communication matrix as needed to configure an existing L2/L3 solution are described in the different chapters and Appendices of this document. An example-architecture which might be used as a general template is defined in the slideset “HAPF20 blueprint”.

5.3.1 VLANs and virtual IP addresses

As depicted in Figure 4: *Required LAN Interface assignment/cabling to traffic categories on Netra X4270 servers* on page 12, interfaces eth1 and eth2 will be connected to a bonding interface and several VLANs will use this “trunk”.

5.3.1.1 Bonding mode

Nature of the bonding connection can be configured if needed. In the default setting of the installation, `eth1` will be active and `eth2` a cold standby. This way, minimum requirements toward a Layer 2 solution exist. Explicitly said: It is not necessary, that used L2 switches support link aggregation protocols. There is further no risk of L2 loops, hence, no Spanning-tree setup is mandatory. In case 802.3ad link aggregations are required, the bonding driver can be configured accordingly. This would happen after the installation by manual changing the bonding options in `/etc/modprobe.conf`.

© 2010 Nokia Siemens Networks. All rights reserved.

The only requirement towards an existing L2 solution is that it must support 802.1q tags towards the interfaces eth1 and eth2 on the servers.

5.3.1.2 CA- traffic trunk

As CA traffic, the following is categorized and will be transported only via the trunk on eth1/eth2 (i.e. bond0):

- TLS traffic between the FE and BE active nodes. That is for INSTA's internal communication
- EE traffic towards the CA. This can include various services to be configured under INSTA, and each service can be mapped to a socket (i.e. IP address : tcp-port, see also Figure 8).
- LDAP traffic between the BE and the FE, with write-permissions on the FE, in case LDAP publishing is used.

IP addresses for different traffic types should be in different VLANs, so that a Firewall can separate the traffic that enters the RAN Security Zone and also internal TLS traffic between Frontend/Backend, as illustrated in Figure 7 above. That means, at least the IPs for external (Zone 2) and internal (Zone 3) shall be different on FEs.

Better traffic separation is achieved, if each service that crosses the RANZS has an own IP address on the FE servers, this is referred here as "vertical" traffic separation. For example, CMP, SCEP, LDAP etc services available on the FEs should all have own IP addresses in separate VLANS.

5.3.1.3 VLAN1ID and internal TLS traffic

In the installation parameters excel sheet, VLANs, physical IPs and virtual HA IP addresses are defined for the traffic in the bonding interface. A special meaning is given here to the first VLAN, with parameter name `VLAN1ID`. While all the other VLANs are highly recommended, but still not mandatory to execute the setup procedure, VLAN1ID is required from the beginning.

It is used to carry TLS and secure LDAP traffic between the BE and FE servers. During installation, the FE servers must contact the BE via this interface in order to enroll a certificate for internal communication. The same certificate can be used for both, certifier internal message exchange and LDAP publishing towards the FE LDAP servers.

See also section 5.3.1.5 "*General IP* Redundancy"

5.3.1.4 Mapping INSTA Services to IP addresses

As depicted below, mapping of INSTA services to IP addresses can be configured in the admin GUI. Within the settings of each individual service, a socket (Bind Address) is specified. This must be a free (local) IP address and port to which the service will connect at startup, no connection attempts to other than the defined Socket will be accepted.

© 2010 Nokia Siemens Networks. All rights reserved.

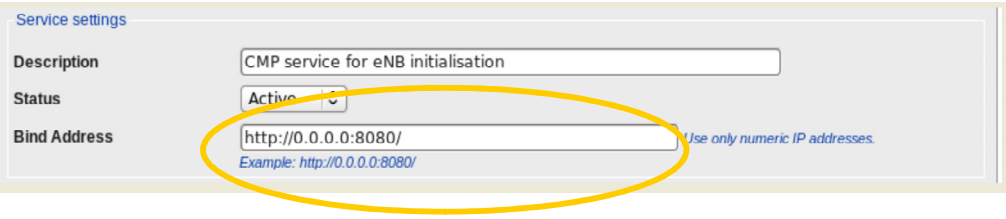


Figure 8: *Mapping of Services to Sockets in the INSTA GUI must match the VLAN/IP configuration*

© 2010 Nokia Siemens Networks. All rights reserved.

5.3.1.5 General IP Redundancy Concept

Redundancy of both, INSTA external services and IPs for internal communication is the established by Linux HA using *corosync* and a proper VLAN/IP configuration:

Generally, one VLAN should be used per external service. For example, CMP, SCEP and WebEnrollment should be in separate VLANs and all those VLANs will be connected via the bond0 trunk. However, besides the fact that one VLAN is always required for internal BE-FE traffic, there is no fixed rule for grouping or for the total number of VLANs. Just because there is no rule which services will be configured. Usually, most services will be towards EEs and run on the FE's. On BE servers, probably only a VLAN for internal communication is needed and eventually a separate one for the PKI admin service.

For each VLAN, each server of an HA pair will have a physical IP address assigned. Additionally, heartbeat manages a third, virtual address of each VLAN and maintains it active only on the currently active machine. Despites differences in the protocol/functionality, this mechanism is the same as known from e.g. HSRP/VRRP or other clustering products like MC/Service Guard or SUN Cluster.

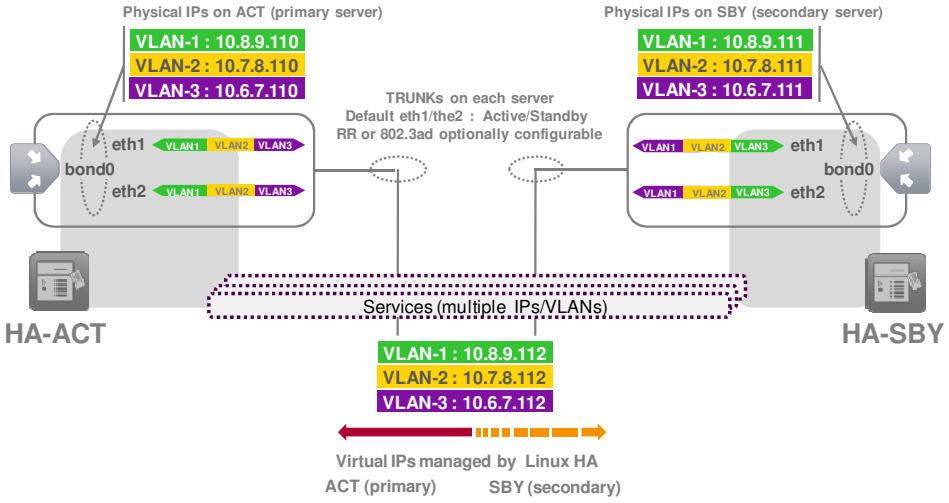


Figure 9: *LAN interface usage and physical vs. virtual IP addresses*

5.3.1.6 Exception for FE-BE internal communication in VLAN1ID

One exception to the above is made on the FE servers, where no HA-IP is in use towards the Backend machines. In expression, only the external service IPs are highly available. Communication towards the BE is initiated by both FE servers at the same time and the BE will reply to the physical addresses. This way it is ensured, that in a longer time without HA-failover the TLS certificate for internal communication will not expire.

See also the paragraph 5.5 and explanations about the FE HA principle..

© 2010 Nokia Siemens Networks. All rights reserved.

5.3.2 Horizontal traffic separation

For the purpose of this document, we call "*horizontal traffic separation*" the filtering of traffic "from left to right", so from RAN via the FEs towards the BEs. Independent of the number of (vertical) VLANs created, a horizontal chain of firewall filters adds additional security. Horizontal traffic separation builds "layers" of access to the CA and is the reason of having separated FE and BE servers in a full deployment. With 2 or 6 servers (i.e. 1 or 2 FE pairs) different scenarios are possible. Recommended and supported is only one scenario for HA deployment, with one vertical VLAN per service in Zone 2 (and also in Zone 1 consequently), and 2 additional VLANs, in one in Zone 3 and one in Zone 4, as depicted in Figure 7 above.

5.3.3 Redundant HA link

All HA and replication traffic is directed via the point-to-point link eth3 that connects the 2 peers of each cluster. In order not to create a single point of failure on this link, it is necessary, to provide an alternative channel for the cluster communication.

The OAM interface eth0 has been configured as a second "ring" in the TOTEM protocol for corosync to offer this redundant channel. This implies, that UDP multicast, the DRBD ports and HSM synchronization must be possible also via the OAM network.

See also the corosync configuration in Figure 25.

© 2010 Nokia Siemens Networks. All rights reserved.

5.4 Network configuration settings

Following Paragraphs describe the files and parameters to enable networking in the different deployment scenarios

5.4.1 Single Server network configuration

Networking on the Single Server can use either 1 or 3 physical (or vmware virtual) network interfaces. The setup procedure will recognize, if a `VLAN1ID[0]` parameter has been defined for the Single Server role. If this is the case, it automatically assumes 3 interfaces to be present. All other settings –loading of drivers or IP addresse assignment, works the same way as it does for the other HA setups.

MIND: If only 1 network interface is present for the Single Server, but at the same time `VLAN1ID` is configured in the planning sheet, the installation procedure will end with an error message!

5.4.2 HA setup network configuration

Simple HA (2BE servers) or any other full HA setup always use a bond0 and a total of 4 network interfaces as described above.

5.4.2.1 Enable and configure networking drivers

Following settings to are made in **`/etc/sysconfig/networking`**

```
alias eth0 e1000
alias eth1 e1000
alias eth2 e1000
alias eth3 e1000
options ipv6 disable=1
alias vlan 8021q
alias bond0 bonding
options bond0 mode=active-backup miimon=100
```

Figure 10: *Network driver initialization settings in modprobe*

Following settings to configure networking drivers are made in **`/etc/modprobe.conf`**

```
alias eth0 e1000
alias eth1 e1000
alias eth2 e1000
alias eth3 e1000
options ipv6 disable=1
alias vlan 8021q
alias bond0 bonding
options bond0 mode=active-backup miimon=100
```

Figure 11: *Network driver initialization settings in modprobe*

© 2010 Nokia Siemens Networks. All rights reserved.

5.4.2.2 Configure Network Interfaces

Following configuration settings are made in `/etc/sysconfig/network-scripts`

```
DEVICE=lo
IPADDR=127.0.0.1
NETMASK=255.0.0.0
NETWORK=127.0.0.0ONBOOT=yes
```

Figure 12: *Definitions for loopback interface in* ***`ifcfg-lo`***

```
DEVICE=eth0
BOOTPROTO=static
HWADDR=[00:0C:29:01:56:EE]
IPADDR=$(IPADDReth0[X])
NETMASK=${NETMASKeth0[X]}
ONBOOT=yes
```

Figure 13: *Definitions for physical interface eth0 in* ***`ifcfg-eth0`*** *(MAC-address dynamically from setup)*

```
DEVICE=eth1
BOOTPROTO=none
MASTER=bond0
SLAVE=yes
HWADDR=[00:0C:29:01:56:F8]
onboot=yes
MTU=1431
```

Figure 14: *Definitions for physical interface eth1 in* ***`ifcfg-eth1`*** *(MAC-address dynamically from setup). Settings for* ***`ifcfg-eth2`*** *are identical, only* ***`HWADDR`*** *and* ***`DEVICE`*** *naturally differ.*

```
DEVICE=eth3
BOOTPROTO=static
NETMASK=$(HBIPMASK[X])
ONBOOT=yes
IPADDR=${HBIPADDR[X])
MTU=1431
```

Figure 15: *Definitions for physical interface eth3 in* ***`ifcfg-eth3`*** *(MAC-address dynamically from setup)*

```
DEVICE=bond0
BOOTPROTO=none
ONBOOT=yes
BONDING_OPTS="mode=1"
```

Figure 16: *Definitions for bond0 interfcae in* ***`ifcfg-bond0`***

© 2010 Nokia Siemens Networks. All rights reserved.

```
DEVICE=bond0.${VALN1ID[X]} (or VLAN2ID[X], …)
BOOTPROTO=none
VLAN=yes
ONBOOT=yes
MTU=1400
IPADDR=${VLAN1IPADDR[X]} (or VLAN2IPADDR[X], …)
NETMASK=${VLAN2MASK[X]} 9or VLAN2MASX[X], …)
```

Figure 17: *Definitions for VLAN interface. There will be one* **`ifcfg-bond0.[X]`** *for every* **`[X=VLAN ID]`** *where* **`X`** *is any of the VLAN IDs defined in the planning sheet*

5.4.2.3 Definition of static routes

Following settings are defined to **`/etc/sysconfig/static-routes`**

```
any net ${NET1[$X]} netmask ${NETMASK1[$X]} gw ${NETGW1[$X]}
...
any net ${NET5[$X]} netmask ${NETMASK5[$X]} gw ${NETGW5[$X]}
```

Figure 18: *Definitions of static routes for all interfaces, in* **`static-routes`**. *Each static route from the planning sheet (marked there as NETX, X=1…5) and corresponding netwmask and gateway will be represented by one route entry of above structure.*

© 2010 Nokia Siemens Networks. All rights reserved.

5.4.2.4 Define hostname resolution

Following settings are defined to **/etc/hosts**

```
127.0.0.1 localhost localhost.localdomain
${IPADDReth0[1]} ${HOSTNAME[1]} fe1
${IPADDReth0[2]} ${HOSTNAME[2]} fe2
${IPADDReth0[3]} ${HOSTNAME[3]} be1
${IPADDReth0[4]} ${HOSTNAME[4]} be2
${HBIPADDR[3]} ${HOSTNAME[3]}-ha
${HBIPADDR[4]} ${HOSTNAME[4]}-ha
```

Figure 19: *Definitions for* ***/etc/hosts*** *on all servers in a PKI. Mind an alias for the role.*

Depending on whether a DOMAIN[X] has been specified in the planning sheet or not, either DNS resolution is set to this domain or switched off, respectively.

Following settings are defined to **/etc/nsswitch.conf** if no domain is configured

```
hosts:      files
```

Figure 20: *Definitions for hostname resolution in* ***nsswitch.conf***

Following settings are defined to **/etc/resolv.conf** if a domain is configured. Further, the nsswitch.conf will also keep its default entry for DNS.

```
nameserver ${NAMESERVER[$X]}
domain ${DOMAIN[$X]}
```

Figure 21: *Definitions for hostname resolution in* ***/etc/resolv.conf***

© 2010 Nokia Siemens Networks. All rights reserved.

5.5 HA Solution Principle

Current release of the HA Platform HAPF2.0 uses 2 separate mechanisms to ensure full functionality of a HA peer:

- Linux HA via corosync/pacemaker controls IP failover and disk replication failovers. Further, Insta services are supervised via an own Resource agent `ICertifier` that checks the availability of all single Insta certifier processes. All resources on a host are grouped together via so-called pacemaker `colocation` constraints and `order` constraints. If any resource in the group fails, the entire group will perform a switchover. State-changes of cluster resources are informed by SNMP trap notifications.

- SNMP based process supervision provides an additional HA layer for local OS processes that can not profit from `corosync` failover capacities. If any of the vital OS processes that have been configured to SNMP based supervision crashes on a host, `net-snmpd` will restart it and send an alarm trap. SNMPD itself is supervised by a cron-based mechanism, while cron in turn is supervised by net-SNMP. This way, a single failure of a vital process will always be recovered automatically.

5.5.1 Linux HA configuration

HAPF2.0 installs so-called “Linux-HA” comprising `corosync` clustering software and `pacemaker` resource manager. Via `pacemaker` configuration, the HA resources are bundled into “*groups*” and become accessible for management via the `crm` resource manager shell.

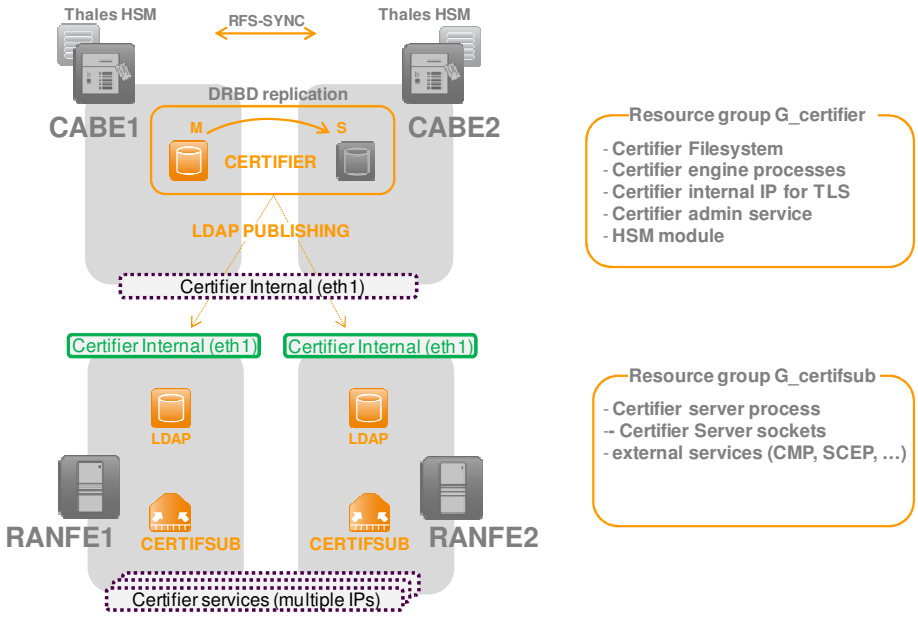


Figure 22: *Illustration of resource groups as used in a full-HA setup (2BE-2FE)*

© 2010 Nokia Siemens Networks. All rights reserved.

Mind, that in the above picture, the concept of “groups” does only factually exist, but not as an “address” for all resources. Definitions in HAPF2.0 are made explicitly to single resources a group is created by `colocation` and `order` statements. Therefore, resources behave as a group and when one single resource is moved to the failover node, all others will follow. There is no configuration made which would allow any single resource to be independent (i.e. if one single IP address on a FE becomes unavailable, the entire internal communication, certifier-server and all other IPs will switch over.

There an automatic fallback to a default node after failure recovery (preferred location).

5.5.1.1 Linux Clustering Software

On HA setups (2+ servers), runlevel 4 triggers an rc-init script to start HA services:

```
/etc/init.d/corosync [start|stop|status|restart|reload|force-reload]
```

In order to restart the entire clustering functionality, the `corosync` init script should be used (simultaneously on both machines in a HA pair). Resources under cluster supervision can be monitored and managed using the `crm_mon` command. In order to switch over the resource group without restarting the cluster, the resource manager command `crm resource migrate` should be used.

Once `corosync` daemons are running, the resource agents monitor the health of the certifier resource Groups on each HA pair and according to `pacemaker` configuration parameters. Content of the resource groups is somewhat dynamic, since e.g. the number and ID of used VLANs for vertical traffic separation on a FE is customer specific.

5.5.1.2 Disk replication using DRBD

DRBD resources are managed via the Linbit ® DRBD resource agent. That means, if the cluster is down, the DRBD drive will have been unmounted. In order to access the Certifier disk resources on the BE while the `corosync` cluster is not operational, mount the `drbd1` device manually to `/usr/local/certifier`.

On Frontend machines, DRBD is not used because the only stateful data held there is in the LDAP database files, which are replicated inside LDAP. Hence, DRBD is installed on all machines, but configured and activated only on the BE

5.5.1.3 Insta Certifier Resource agent

Certifier resources under supervision of `corosync` are started/stoped and monitored using the *ICertifier OCF resource agent*. This means the rc-init script should not be used because it would collide with actions taken independently and automatically by `corosync`. However, while the cluster is down and maintenance operations are being carried out, it is “handy” to use the rc-init scripts for certifier in situations where no pacemaker is controlling it. For example, they can be used to quickly start certifier in debug mode or – even while pacemaker is up – to manually control the status.

© 2010 Nokia Siemens Networks. All rights reserved.

ICertifier is called by pacemaker and has only one command line option: Either “be” if it runs on the Backends, or “fe” if it runs on the FEs. In the Backends, a normal active/passive HA is created this way. In the FEs, it would be theoretically possible to have certifier services active on both machines, since the certifier backend application itself supports multiple active FE connections on one BE. However, the HA is per default configured in a way, that this option is NOT used and ICertifier will failover all services – certifier-server and all configured virtual IP addresses – to the standby FE.

5.5.2 SNMP based Process Supervision

In the planning sheet, the `SNMPENA[X]` parameter defines how `net-snmp` should be used. If `SNMPENA[X]` is set to “*yes*”, failure notifications are sent to an external OSS and the server can be polled via SNMP port UDP:161. If `SNMPENA[X]` is set to “*no*”, this functionality is not available but still a rudimentary, local configuration remains to allow supervision of vital system processes. This does also apply for the SingleServer, where no pacemaker failover can be provided, but still OS resources are supervised and auto-recovered with net-snmp.

The SNMP “*procfail*” mechanism is used for this purpose. It reacts on failed processes and triggers a sequence to initiate recovery. If a supervised process is found not running, `snmpd` sets a “`procfail`” flag in its internal MIB tables and sends a failure notification before it executes a so-called “`procfix`” command which must be defined separately for each supervised process.

In order to make sure that `snmpd` itself does not fail to run, a “wrapper script” is called via cron and checks, if `snmpd` is still alive clean (i.e. no leftovers are found in `/var/run`). If it is not running and at the same time it’s PID-files or status/lock files have not been cleaned, it is assumed to have crashed and will restart.

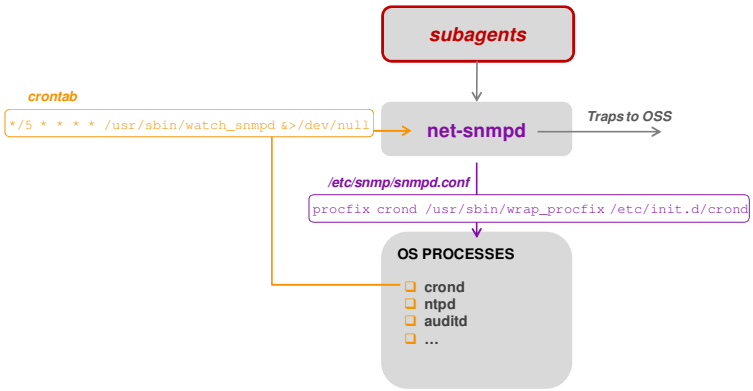


Figure 23: *Schema on mutual supervision of snmpd via cron and vice-versa to prevent SPOF*

© 2010 Nokia Siemens Networks. All rights reserved.

5.5.2.1 Wrappers for snmpd

A shortfall in the described process supervision is, that a service that has been stopped for regular maintenance reasons would be automatically restarted by snmpd. Or – under certain and unlikely rare timing conditions - a process that is running clean but in a debug mode could be restarted by snmpd and then continue in normal mode. In many situations, automatic but unsolicited restart might have an undesirable or damaging effect. To avoid it, the procfail directives are not bound directly to the rc-init scripts to stubbornly restart the processes, but to a wrapper script.

The /usr/bin/wrap_procfix script does a verification of the process PID and its status files under /var/run. Only if a mismatch is found – e.g. the process is running under a different PID than its PID file shows – a restart will be triggered.

5.6 HA configuration settings

To configure certifier HA, `corosync/pacemaker` and DRBD disk replication must be set up.

DRBD is physically a kernel-module placed between the disk resources filesystem drivers (i.e. a partition) and networking drivers. It has an own configuration interface with config files for the module startup and behavior plus a CLI suite. It manages “stateful disks” that have a master state on one node and a slave state for replication on another node. The master/slave state again is managed as “*just another resource*” via `pacemaker` interface by `corosync`. As such, `corosync` decides where the certifier filesystem is mounted and the BE functionality is running.

Basic configuration of the `corosync` daemons goes via own configuration files, where the HA resource manager `pacemaker` provides an independent, shell-type interface to supply `corosync` with triggers to node switchovers. Configuration of all HA resources like IP addresses or processes and DRBD states is made via pacemaker’s CLI invoked by the `crm` command and stored in a `pacemaker` internal database.

Pacemaker fetches its information via resource agent scripts, its configuration files and particular settings in NSN INSTA HAPF2.0. Refer to the documentation available at www.clusterlabs.org to get first-hand information on the functionality of `pacemaker` and `corosync`.

5.6.1 Configuring corosync

`Corosync` daemons are responsible to react upon the status of resources as reported by the `pacemaker` resource manager, which in turn uses dedicated resource agents per each resource to monitor.

© 2010 Nokia Siemens Networks. All rights reserved.

5.6.1.1 Settings in corosync

Following Configuration is applied in `/etc/corosync/corosync.conf`

```
compatibility: whitetank
totem {
                version: 2
                secauth: off
                threads: 0
                        interface {
                             ringnumber: 0
                             bindnetaddr: ${MYNET}
                             mcastaddr: ${HBMCADDR[$X]}
                             mcastport: ${HBMCPORT[$X]}
                }
                        interface {
                             ringnumber: 1
                             bindnetaddr: ${IPaddreth0[$X]}
                             mcastaddr: ${HBMCADDR[$X]}
                             mcastport: ${HBMCPORT[$X]}
                }
}
logging {
                fileline: off
                to_stderr: yes
                to_logfile: yes
                to_syslog: yes
                logfile: /tmp/corosync.log
                debug: off
                timestamp: on
                logger_subsys {
                             subsys: AMF
                             debug: off
                }
}
aisexec {
                 user: root
                 group: root
}
service {
                 name: pacemaker
                 ver:  0
}
```

Figure 24: *Configuration file for `corosync` and enabling of `pacemaker` resource manager functionality. Mind that ${MYNET} is dynamically determined as the network address of `${HBMCADDR[$X]}` by the auto-setup script.*

© 2010 Nokia Siemens Networks. All rights reserved.

5.6.2 Configuring Pacemaker

5.6.2.1 Settings in pacemaker (crm) at the BE servers

Verify following pacemaker settings the "`crm configure show`" as user root.

```
${HOSTNAME[$X]}
${HOSTNAME[$Y]}
primitive certifier_engine ocf:nsn:ICertifier \
        params certifier_role="be" \
        op monitor interval="90s" timeout="30s" \
        op start interval="0" timeout="90s" \
        op stop interval="0" timeout="120s"
primitive drbd_certifier ocf:linbit:drbd \
        params drbd_resource="certifier" \
        op start interval="0" timeout="240s" \
        op stop interval="0" timeout="100s" \
        op monitor interval="25s" role="Master" \
        op monitor interval="35s" role="Slave"
primitive fs_certifier ocf:heartbeat:Filesystem \
        params device="/dev/drbd1" directory="/usr/local/certifier"
fstype="ext3" \
        op start interval="0" timeout="60s" \
        op stop interval="0" timeout="60s" \
        op monitor interval="65s" timeout="30s"
primitive ${VLAN1ID[$X]} ocf:nsn:IPaddr2a \
        params ip="${VLAN1IPvirtual[$X]}" nic="bond0" cidr_netmask="24" \
        op monitor interval="60s" timeout="20s"
group G_certifier fs_certifier ${VLAN1ID[$X]} certifier_engine
ms ms_certifier drbd_certifier \
        meta master-max="1" master-node-max="1" clone-max="2" clone-node-
max="1" notify="true"
location prefer-be1 G_certifier 100: ${HOSTNAME[$X]}
colocation fs_on_drbd inf: G_certifier ms_certifier:Master
order fs_after_drbd inf: ms_certifier:promote G_certifier:start
property $id="cib-bootstrap-options" \
       dc-version="1.1.6-3.el6-a02c0f19a00c1eb2527ad38f146ebc0834814558"\
        cluster-infrastructure="openais" \
        expected-quorum-votes="2" \
        no-quorum-policy="ignore" \
        stonith-enabled="false" \
        last-lrm-refresh="1341594844"
```

Figure 25: *HA parameters defined in the CIB of BE servers. Mind, that additional VLANs could be defined on top of ${VLAN1ID[$X]}, and that they would be included into the group G_certifier.*

© 2010 Nokia Siemens Networks. All rights reserved.

5.6.2.2 Settings in pacemaker (crm) at the FE servers

Below settings show the pacemaker configuration in the “crm configure show” output, where customer-specific values have been replaced here by the name of the parameter from the excel sheet.

```
${HOSTNAME[$X]}
${HOSTNAME[$Y]}
primitive certifsub_server ocf:nsn:ICertifier \
        params certifier_role="fe" \
        op monitor interval="60s" timeout="30s" \
        op start interval="0" timeout="60s" \
        op stop interval="0" timeout="90s"
primitive ${VALN2ID[$X]} ocf:nsn:IPaddr2a \
        params ip="${VLAN2IPvirtual[$X]}" nic="bond0" cidr_netmask="24" \
        op monitor interval="60s" timeout="20s"
group G_certifsub ${VALN2ID[$X]}
clone certifsub_service certifsub_server \
        meta clone-node-max="1" globally-unique="false" target-
role="Started"
location prefer-fe1 G_certifsub 100: ${HOSTNAME[$X]}
order rebind_service inf: G_certifsub:start certifsub_service
property $id="cib-bootstrap-options" \
       dc-version="1.1.6-3.el6-a02c0f19a00c1eb2527ad38f146ebc0834814558"\
       cluster-infrastructure="openais" \
       expected-quorum-votes="2" \
       stonith-enabled="false" \
       no-quorum-policy="ignore"
```

Figure 26: *HA parameters defined in the CIB of FE servers. Mind, that additional VLANs could be defined on top of ${VLAN2ID[$X]}, and that they would be included into the group G_certifsub. Mind, that in case of BE servers, additional VLANs are rather unlikely. Whereas in case of FE servers, it is very likely that more than one VLAN is used to provide IPs for the external services.*

© 2010 Nokia Siemens Networks. All rights reserved.

5.6.3 Configuring DRBD

In order to allow disk replication between BE servers, Linbit® Duplicated Replicated Block Device (DRBD) is used. Only BE role servers use DRBD to replicate the certifier database and settings made inside the filesystem mounted under `/usr/local/certifier`

5.6.3.1 Global Settings in /etc/drbd.d/global_common.conf

Following is the central configuration for global parameters of disk replication

```
global {
          usage-count no;
}
common {
          protocol C;
          handlers {
          }
          startup {
                    degr-wfc-timeout 35;
                    wfc-timeout 15;
          }
          disk {
                    on-io-error detach;
          }
          net {
                    allow-two-primaries yes;
                    after-sb-0pri discard-younger-primary;
                    after-sb-1pri consensus;
                    after-sb-2pri disconnect;
          }
          syncer {
          }
}
```

Figure 27: *Settings for disk replication in* ***`/etc/drbd.d/global_common.conf`***

© 2010 Nokia Siemens Networks. All rights reserved.

5.6.3.2 Resource specific settings in /etc/drbd.conf

Following is the configuration for resource specific parameters of disk replication

```
include ${RESFIL};
resource ${DRBDRES} {
                    device ${DRBDDEV};
                    disk ${PDRBD};
                    meta-disk internal;
                    syncer {
                             rate 40M;
                             verify-alg crc32c;
                    }
                    net {
                    }
                    on ${HOSTNAME[$X]} {
                                  address ${HBIPADDR[$X]}:${DRBDPRT};
                    }
                    on ${HOSTNAME[$Y]} {
                                  address ${HBIPADDR[$Y]}:${DRBDPRT};
                    }
}
```

Figure 28: *Settings for disk replication in* ***`/etc/drbd.conf`****. Mind, that the variables ${DRBDDEV}, ${PDRBD} and ${RESFIL} are dynamically set by the auto installation scripts and are not subject to customizations in the planning sheet.*

Mind, that the DRBD resources must be allowed on eth0 as well as on eth3 (see also paragraph 5.3.3)

© 2010 Nokia Siemens Networks. All rights reserved.

5.7 Configuring snmp

In HAPF2.0, snmpd is used for regular OAM purposes as well as for process supervision and recovery. As described in previous paragraphs of this chapter, it triggers fault notifications about OS processes and restarts OS processes once they crashed. The next paragraph 0 explains the OAM integration aspects. Below here, the actual configuration file is shown.

```
# We recommend you to remove the lines containing createUser directives
# Those are only needed for clean initialisation and can be removed later
# USM info is stored persistant and encrypted in /var/net-snmp/snmpd.conf
engineID ${MYENGID}
###############################################################################
agentaddress ${IPADDReth0[X]}
leave_pidfile no
syslocation ${SYSLOC[$X]}
sysContact ${SYSCON[$X]}
maxGetbulkRepeats 64
maxGetbulkResponses 64
###############################################################################
rocommunity ${SNMPV2COMMUNITY[$X]}
###############################################################################
createUser pkisuper  SHA pkisuper_passphrase AES pkisuper_passphrase
createUser ${SNMPUSER} SHA ${SNMPV3PASSPHRASE[$X]} DES
rwuser internal
###############################################################################
com2sec pkisuper default SNMPV2COMMUNITY[$X]}
group redhat usm pkisuper
group pki    usm ${SNMPUSER}
#
view all         included .iso.org.dod.internet
view mib2        included .iso.org.dod.internet.mgmt.mib-2
view mib2        included .iso.org.dod.internet.private.enterprises
view insta       included .iso.org.dod.internet.private.enterprises.insta
access redhat     "" usm priv exact mib2  none mib2
access pki        "" usm auth exact insta none insta
###############################################################################
# Host Resources MIB
#
proc crond
procfix crond /usr/sbin/wrap_procfix /etc/init.d/crond
#
proc auditd
procfix auditd /usr/sbin/wrap_procfix /etc/init.d/auditd
#
proc /sbin/rsyslogd
```

© 2010 Nokia Siemens Networks. All rights reserved.

```
procfix /sbin/rsyslogd /usr/sbin/wrap_procfix /etc/init.d/rsyslog
#
proc /usr/sbin/sshd
procfix /usr/sbin/sshd /usr/sbin/wrap_procfix /etc/init.d/sshd
#
proc ntpd
procfix ntpd /usr/sbin/wrap_procfix /etc/init.d/ntpd
#
disk / 30%
disk /backup 30%
disk /usr/local/certifier 30%
#
##############################################################################
trapcommunity ${SNMPV2COMMUNITY[$X]}
trap2sink ${SNMPTRAPRCVIP[$X]}
agentSecName internal
#
#FIRES:
# trap 1.3.6.1.2.88.2.0.1 (MTE trigger) with below -o add varbinds:
# 1.3.6.1.4.1.2021.2.1.2.X is procname & 1.3.6.1.4.1.2021.2.1.101.X
# is text in prErrMessage X=1..6
monitor -r ${PROCMT} -e cronDead   -i -o prNames.1 -o prErrMessage.1
"cronProcCheck"    prErrorFlag.1 != 0
monitor -r ${PROCMT} -e auditDead  -i -o prNames.2 -o prErrMessage.2
"auditProcCheck"   prErrorFlag.2 != 0
monitor -r ${PROCMT} -e syslogDead -i -o prNames.3 -o prErrMessage.3
"syslogProcCheck"  prErrorFlag.3 != 0
monitor -r ${PROCMT} -e sshDead    -i -o prNames.4 -o prErrMessage.4
"sshProcCheck"     prErrorFlag.4 != 0
monitor -r ${PROCMT} -e ntpDead    -i -o prNames.5 -o prErrMessage.5
"ntpProcCheck"     prErrorFlag.5 != 0
#monitor -r ${PROCMT} -e ldapDead   -i -o prNames.6 -o prErrMessage.6
"ldapProcCheck"    prErrorFlag.6 != 0
#
setEvent cronDead    prErrFix.1 = 1
setEvent auditDead   prErrFix.2 = 1
setEvent syslogDead  prErrFix.3 = 1
setEvent sshDead     prErrFix.4 = 1
setEvent ntpDead     prErrFix.5 = 1
#setEvent ldapDead    prErrFix.6 = 1
#
notificationEvent cronDead    mteTriggerFired
notificationEvent auditDead   mteTriggerFired
notificationEvent syslogDead  mteTriggerFired
notificationEvent sshDead     mteTriggerFired
notificationEvent ntpDead     mteTriggerFired
#notificationEvent ldapDead    mteTriggerFired
#
monitor -r ${DSKMT} -e fsFull  -o dskPath -o dskErrorMsg "fsCheck"
```

© 2010 Nokia Siemens Networks. All rights reserved.

```
dskErrorFlag != 0

monitor -r ${DSKMT} -e fsClear -o dskPath               "fsClear"
dskErrorFlag == 0

notificationEvent fsFull mteTriggerFired

notificationEvent fsClear mteTriggerFired

#

monitor -s -r ${LNKMT} -e linkUpTrap   "Generate linkUp"   ifOperStatus
!= 2

monitor -s -r ${LNKMT} -e linkDownTrap "Generate linkDown" ifOperStatus
== 2

notificationEvent  linkUpTrap    linkUp   ifIndex ifDescr ifAdminStatus
ifOperStatus

notificationEvent  linkDownTrap  linkDown ifIndex ifDescr ifAdminStatus
ifOperStatus

g#

###############################################################################
```

Figure 29: *Settings for ne-snmp in* ***`/etc/snmp/snmpd.conf`****. Mind, that the monitoring timings can be set before the installation inside the install scripts..*

The above SNMP configuration shows only one possible example, additional processes might be added for local monitoring or different settings might be applied for the way how the engine-ID is set. Those details depend on the installation definitions made in the excel sheet.

© 2010 Nokia Siemens Networks. All rights reserved.

5.8 Preparing OAM Integration

Following functions related to external OAM systems are provided by net-snmp in NSN INSTA HAPF2.0

- Ability to poll PM measurements in SNMPv3 and SNMPv2c from the servers
- Ability to monitor OS events and send according SNMPv3/v2c fault notifications.
- Ability for the backend engine, to send SNMPv3 traps
- Built-in ability of net-snmptrapd, to convert the backend applications SNMPv3 traps into SNMPv2c traps that fit the NetAct SNMP adaptation toolkit.

In order to allow a plug&play adaptation with NSN NetAct (or any other desired OSS), the SNMPv3 traps sent from the certifier engine might require a conversion onto SNMPv2. This is because some network management systems still do not fully support SNMPv3 type notifications. For this case, a conversion of traps as described in the following can be done, after certifier SNMPv3 notifications are enabled according to the description in the next paragraph.

5.8.1 Settings in certifier engine.conf

Sending of fault notifications related to the Insta certifier functions is handled by Insta's own certifier-snmp-daemon. It allows certain configuration options for the sending of SNMPv3 notifications see the Insta OSS Guide for details.

Certifier-engine does naturally send notifications in SNMPv3 traps format. In order for this to be understood by any trap handler, the SNMP engine-ID must be known there! The configuration parameter `ENGID` in the planning sheet can be set to "from engine.conf", which means, that the engine-ID in the net-snmp configuration is taken from the `/usr/local/certifier/conf/engine/conf` settings. Alternatively, other methods supported by net-snmp with the "`engineIDType`" directive can also be specified for certifier, but they must always be accordingly in the `engine.conf` and s`nmpd.conf`! If no particular requirements from the OSS demand different, it is highly recommended, to leave the settings in the planning sheet on "*from engine.conf*"

5.8.2 Preparing trap conversion to SNMPv2c

INSTA snmp-daemon sends its FM notifications only as SNMPv3 protocol PDU. In order to allow NetAct or other systems to decode those traps, they must first be converted into SNMPv2c trap-notifications. This conversion-loop is done on the Insta PKI Platform itself and uses net-snmp `snmptrapd`. If sending to SNMPv2c traps exclusively (i.e. conversion) should be enabled, the `SNMPV3CONV[X]` parameter must be set to "*yes*". In that case, all PKI traps from `certifier-snmp-daemon` are sent to `snmptrapd` on the loopback 127.0.0.1, and forwarded by a handler-script as unencrypted SNMPv2c trap-PDU to the IP address specified in the Planning sheet with `SNMPTRAPRCVIP`.

© 2010 Nokia Siemens Networks. All rights reserved.

5.8.2.1 Configuring snmptrapd

Following settings are applied to the `snmptrapd` trap receiver on the PKI, the receiving end of the conversion loop:

```
#############################################################################
# put IPaddr here from where to receive notifications
#snmpTrapdAddr 127.0.0.1
#############################################################################
#doNotRetainNotificationLogs no
#doNotLogTraps no
pidFile /var/run/snmptrapd.pid
# this user must be bound to engine ID as configured to certifier engine.conf
createUser -e "0x80001F8804636572746966696572" certifier SHA certifier_passphrase
authUser log,execute,net certifier
logOption f /var/log/traplog
# following must point to your traphandler script
traphandle default /root/traphandler.sh
```

Figure 30: *Settings for trap receiving in* ***`/etc/snmp/snmptrapd.conf`***

5.8.2.2 Placing the traphandler script

Following is the trap-handler script that fires the SNMPv3 traps from certifier as SNMPv2c trap notifications. Mind, that the information in the script header must be adapted manually to point towards the NetAct adaptation package and port, using the proper remote community string. Per default, the handler sends to port 162.

The script should is placed as executable in **`/usr/local/bin`**

```
#!/bin/sh
#############################################################
# mini-traphandler for Insta certifier FM adaptation
# 27.05.2011 Mpe
#############################################################
trapreceiver="192.168.1.49:162"
v2trapcomnty="-c pki_v2community"
snmpversion="-v 2c"
#############################################################
# uncomment next line if you need a debug log
#exec > /var/log/debug_traphandler 2>&1
read host
read ip
echo -n "$$ `date +%d%m%Y-%H:%m:%S` : "
echo "received something from $host $ip ..."
let i=1
```

© 2010 Nokia Siemens Networks. All rights reserved.

```
vars=
while read oid val
do
        vbind[$i]=${val}
        vboid[$i]=${oid}
        i=$i+1
done

#########################################################

timeticks=${vbind[1]}
trpentoid=${vbind[2]}

varbinds="";let j=3
while [ ${vboid[$j]} ]
do
        echo -n "$$ `date +%d%m%Y-%H:%m:%S` : "
        echo "decoding varbind ${j} : ${vboid[$j]} = ${vbind[$j]}"
        varbinds="${varbinds} ${vboid[$j]} = ${vbind[$j]}"
        let j=$j+1
done
# now shoot:
echo -n "$$ `date +%d%m%Y-%H:%m:%S` : "
echo  "organized data and will fire as follows:"
echo  ${snmpversion} ${v2trapcomnty} ${trapreceiver} ${timeticks} ${trpentoid} ${varbinds}
snmptrap ${snmpversion} ${v2trapcomnty} ${trapreceiver} ${timeticks} ${trpentoid} ${varbinds}
```

Figure 31: *Script to receive and forward traps, thereby doing a conversion from SNMPv3 to SNMPv2c.*

© 2010 Nokia Siemens Networks. All rights reserved.

6. Abbreviations and References

6.1 Abbreviations

Abbreviation	Description
VM	Virtual Machine
HA	High Availablity
EE	End Entity
CA	Certification Authority
PKI	Public Key Infrastructures
HSM	Hardware Security Module
PSK	Pre-Shared key
FE	Frontend
BE	Backend
FE1, FE2	Active (1) and Standby (2) roles in a FE HA pair
BE1, BE2	Active (1) and Standby (2) roles in a BE HA pair
TLS	Transport Layer Security
SPOF	Single Point of Failure

6.2 References

Ref	Description
/1	NSN Insta PKI Installation Guideline for HAPF2.0
/2	LTE Transport Security Sizing & Ordering Guideline
/3	Radio Transport Security Solution - Technical Solution Description
/4	PKI Configuration Quick-Start Guide
/5	HAPF FM PM Reference
/6	HAPF20 Admin Notes (Collection of independent Leaflets)

© 2010 Nokia Siemens Networks. All rights reserved.

Appendix A: Communication Matrix and FW Policy

Following communication ports are usually assigned within the PKI. Mind, that socket mapping of services might create additional needs, also that heartbeat and DRBD ports are configured as described below, but can be changed if needed.

		default port	BE1		BE2		FE1		FE2	
			into	out	into	out	into	out	into	out
PKI internal	Admin Service	http(s):8083	X		X					
	TLS internal	tcp:7001	X	X	X	X	X	X	X	X
Linux HA	DRBD	tcp:7789	X	X	X	X				
	corosync	udp:5400	X	X	X	X	X	X	X	X
	corosync (UC)	udp:5401	X	X	X	X	X	X	X	X
	corosync (MC)	udp:5401	X	X	X	X	X	X	X	X
PKI Services	range	http:8080					(X)		(X)	
	(default)	- from/to -					(X)		(X)	
		http:8090					(X)		(X)	
Validate & Publish	LDAP(S)	tcp:389	X	X	X	X	X	X	X	X
Generic / Operation	SSH	tcp:22	X		X		X		X	
	NTP	udp:123	X		X		X		X	
	SNMP	udp:161	X		X		X		X	
	SNMP Trap	udp:162		X		X		X		X

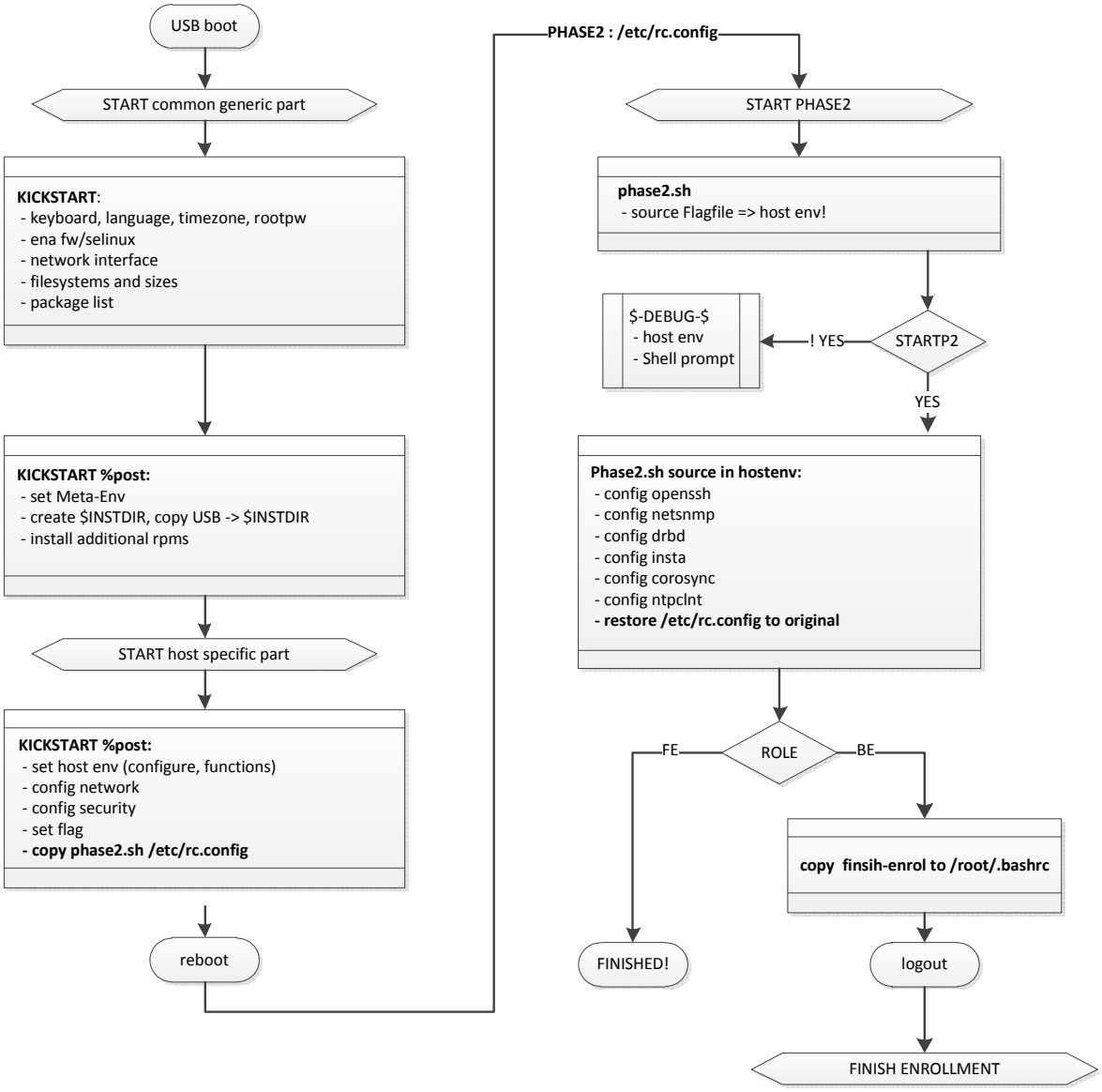
Figure 32: *Default port numbers and (suggested) usage per role, mind that actual implementation may differ*

In the given figure, only PKI internal communication and Linux HA config are predefined after the installation and should not be changed. All other ports depend on the particular settings made to the service in the INSTA GUI.

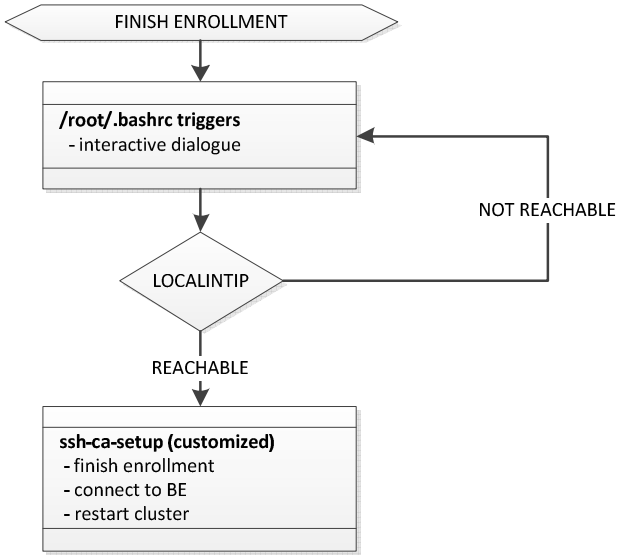
© 2010 Nokia Siemens Networks. All rights reserved.

Appendix B: Flow of actions during rapid-setup

Following charts do roughly depict the flow of actions taken during the rapid-setup procedure. For a full understanding, refer to the shell scripts itself. Start in the order they are appear in the charts below.



© 2010 Nokia Siemens Networks. All rights reserved.



© 2010 Nokia Siemens Networks. All rights reserved.