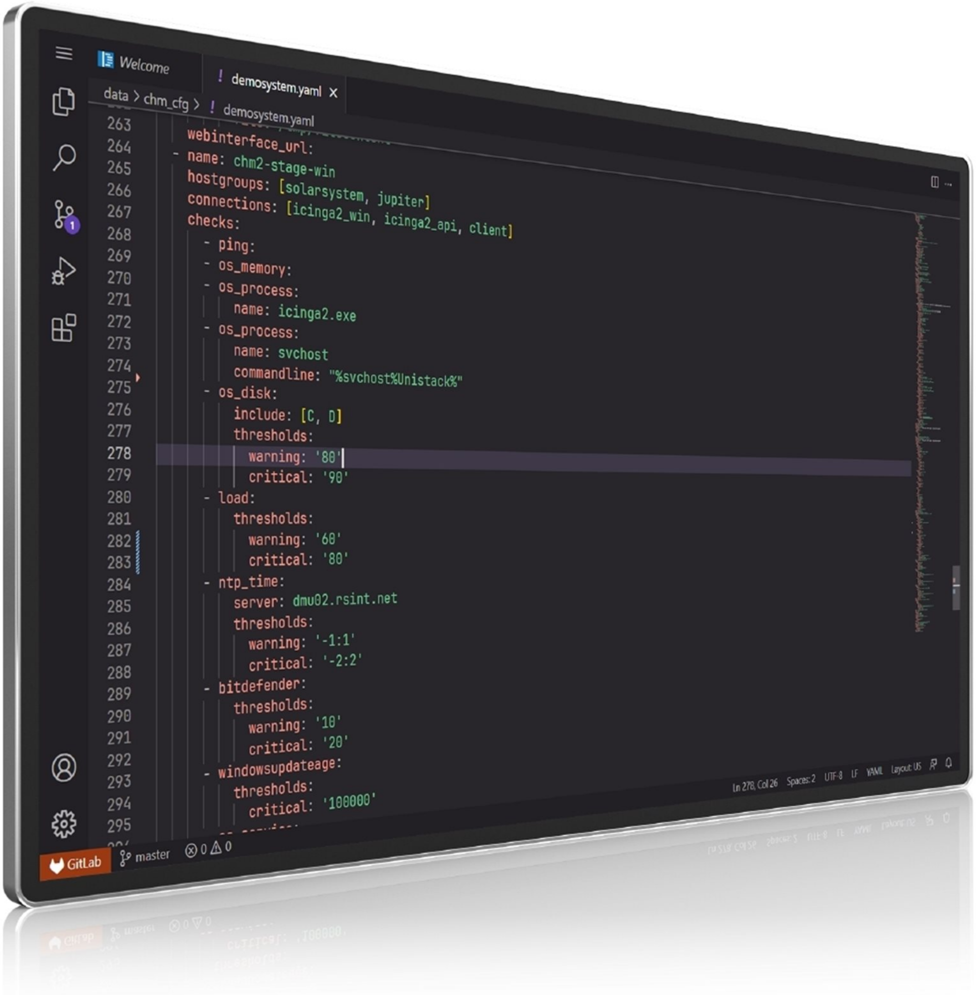
R&S®CHM
System Status Monitoring Configuration
Configuration Manual



```
  webinterface_url:
- name: chm2-stage-win
  hostgroups: [solarsystem, jupiter]
  connections: [icinga2_win, icinga2_api, client]
  checks:
    - ping:
    - os_memory:
    - os_process:
        name: icinga2.exe
    - os_process:
        name: svchost
        commandline: "%svchost%Unistack%"
    - os_disk:
        include: [C, D]
        thresholds:
          warning: '80'
          critical: '90'
    - load:
        thresholds:
          warning: '60'
          critical: '80'
    - ntp_time:
        server: dmu02.rsint.net
        thresholds:
          warning: '-1:1'
          critical: '-2:2'
    - bitdefender:
        thresholds:
          warning: '10'
          critical: '20'
    - windowsupdateage:
        thresholds:
          critical: '100000'
```

1179613702
Version 09

ROHDE&SCHWARZ
Make ideas real

This document describes implementation and configuration of the following software with version 2411 and higher:

- R&S®CHM, system status monitoring software (3067.6545.02)

© 2025 Rohde & Schwarz
Muehldorfstr. 15, 81671 Muenchen, Germany
Phone: +49 89 41 29 - 0
Email: info@rohde-schwarz.com
Internet: www.rohde-schwarz.com
Subject to change – data without tolerance limits is not binding.
R&S® is a registered trademark of Rohde & Schwarz GmbH & Co. KG.
All other trademarks are the properties of their respective owners.

1179.6137.02 | Version 09 | R&S®CHM

Throughout this document, R&S® is indicated as R&S.

Contents

1 Welcome to R&S CHM

The R&S CHM software monitors status information from various system components that are connected to the network. The web-based user interface visualizes system state parameters, and lets you monitor and troubleshoot connected and configured Rohde & Schwarz instruments, devices with simple network management protocol (SNMP) interface, and other hosts.

Target audience

This manual familiarizes you with implementation and configuration of R&S CHM, including configuration of monitoring services. As a **system administrator** or **software integrator**, you install and configure R&S CHM on the R&S CHM host. These configuration tasks require *root* user access. It is assumed that you already have comprehensive knowledge of system setup and configuration.

For information on using the R&S CHM web GUI, see the "R&S CHM System Status Monitoring" user manual.

1.1 Key features

R&S CHM system status monitoring provides the following high-level features:

- Run on a security-enhanced Linux distribution (SELinux)
- Run on a hardened operating system according to DISA STIGs. For information, see https://public.cyber.mil/stigs/.
- Run unattended for a long period of time
- Continuously monitor the status of hosts and services, e.g. used disk space
- Allow configuration of device-specific monitoring services
- Reduce downtime of system components
- Troubleshooting of problems
- Encrypted communication between R&S CHM and monitored hosts
- Secure password handling

1.2 Documentation overview

This section provides an overview of the R&S CHM user documentation. Unless specified otherwise, you find the documents at:

www.rohde-schwarz.com/product/chm

1.2.1 Manuals

The manuals are provided in two formats. The PDF format is contained in the software delivery. An HTML5-based help format is available on the R&S CHM web GUI.

The latest versions of the manuals are available for download or for immediate display on the internet at:

www.rohde-schwarz.com/manual/chm

- **"R&S CHM System Status Monitoring" user manual**:
  Introduces the R&S CHM and describes how to start working with the web GUI that lets you monitor the "health status" of the system in detail.
- **"R&S CHM System Status Monitoring Configuration" configuration manual**:
  Provides a description of all configuration options and describes how you implement and set up R&S CHM on all system components.

To obtain help on the web GUI

1. On the left navigation area of the R&S CHM web GUI, select "Extras" > "User Manual".

   The help opens in the R&S CHM web GUI (English).
2. To show the manual in German, select the "DEU" tab on the top of the "User Manual" area.

1.2.2 Brochure

The brochure provides an overview of the software and deals with the specific characteristics.

See www.rohde-schwarz.com/brochure-datasheet/chm

1.2.3 Release notes and open source acknowledgment (OSA)

The release notes list new features, improvements and known limitations of the current software version, and contain a release history.

The open source acknowledgment document provides verbatim license texts of the used open source software.

Both documents are contained in the software delivery.

2 What's new

This section summarizes the most important changes and enhancements to the documentation. For information about latest product and documentation changes, restrictions and known issues, see the release notes.

2.1 R&S CHM v2411

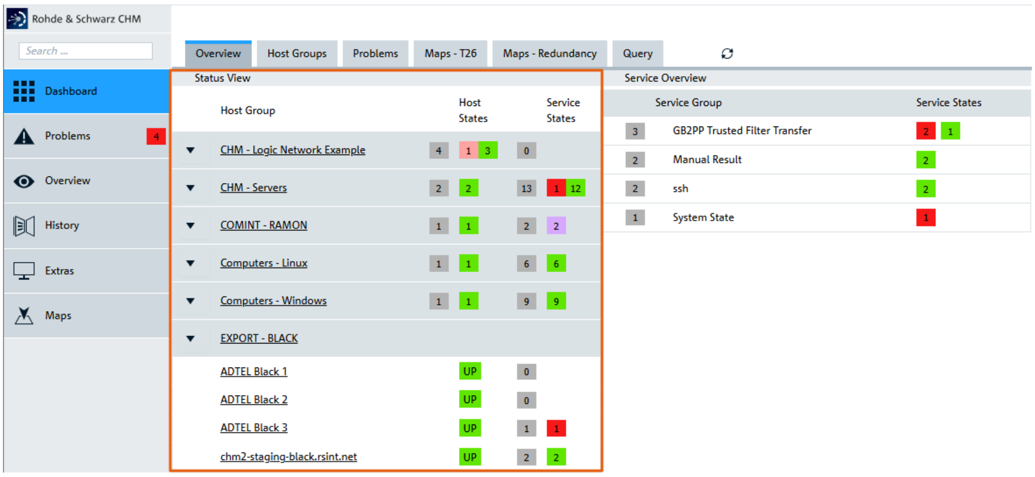
Support of SSO on R&S CHM clients

You can now configure single sign-on on R&S CHM clients to allow users to log in to the web GUI using the Windows user credentials.

Read more: Section 4.3.6, "Setting up SSO", on page 29

Add Status View page to web GUI

You can now add a "Status View" page that provides enhanced grouping functions compared to the built-in "Hostgroups" page. Also, you can add the "Status View" as a widget to the "Dashboard".



Status View

	Host Group	Host States	Service States
▼	CHM - Logic Network Example	4 1 3	0
▼	CHM - Servers	2 2	13 1 12
▼	COMINT - RAMON	1 1	2 2
▼	Computers - Linux	1 1	6 6
▼	Computers - Windows	1 1	9 9
▼	EXPORT - BLACK		
	ADTEL Black 1	UP	0
	ADTEL Black 2	UP	0
	ADTEL Black 3	UP	1 1
	chm2-staging-black.rsint.net	UP	2 2

Service Overview

	Service Group	Service States
3	GB2PP Trusted Filter Transfer	2 1
2	Manual Result	2
2	ssh	2
1	System State	1

Figure 2-1: Status View on the Dashboard

Read more:

- To enable the "Status View" page, see `authorization` on page 63.
- To customize the "Dashboard" page, see `widgets` on page 46.

New or enhanced status checks

- `permissions`
  You can now assign permissions `checkdetails` and `manual` to a web GUI user.
  - The permission `checkdetails` provides more information to the user.

  - The permission manual in combination with the passive check manual lets the user add the status of a device manually.

  Read more: authorization on page 63 > permissions

- trapreceivers
  Configures R&S CHM to inform a list of SNMP notification receivers about system status changes via SNMPv2 traps (optional).
  Read more: Section 6.9.2, "Configuring SNMPv2 traps", on page 77
- manual
  Adds a button for "passive" checks. If there is a device in the system that cannot be monitored, you can enter the manual status here.
  Read more: manual on page 122
- eta_pdu
  Monitors PDUs from ETA that support the MIB eta_RCI11_1.0.1_MIB.mib.
  Check enhancement with new checks of temperature and humidity.
  Read more: eta_pdu on page 111
- lancom_vpn_status
  Monitors the status of VPN connections on a LANCOM device via SNMP.
  Read more: lancom_vpn_status on page 120
- synology
  Monitors various aspects of a Synology NAS via SNMP.
  Read more: synology on page 137

2.2 R&S CHM v2407

Oracle 8 support

R&S CHM can now also run on the operating system Oracle Linux 8.x (and later).

Read more: Section 4.1, "Installing the R&S CHM host without LCSM", on page 22

New or enhanced status checks

- eta_pdu
  Monitors PDUs from ETA that support the MIB eta_RCI11_1.0.1_MIB.mib.
  Read more: eta_pdu on page 111

2.3 Previous releases

2.3.1 R&S CHM v2406

New or enhanced status checks

- mikrotik
  Monitors various aspects of a MikroTik device via SNMP.
  Read more: mikrotik on page 123
- domain
  Monitors a domain.
  Read more: domain on page 110
- gude
  Monitors temperature, humidity sensor and outlets of a Gude power distribution unit (PDU).
  Read more: gude on page 116
- meinberg
  Monitors the network time protocol (NTP) current state and GPS mode for devices that support the MBG-LANTIME-NG-MIB.
  Read more: meinberg on page 122
- SSH
  This check attempts to establish an SSH connection to the specified host and port.
  Read more: ssh on page 136
- navics
  The navics check now can monitor the NAVICS broadcast and alarm system.
  Read more: navics on page 124

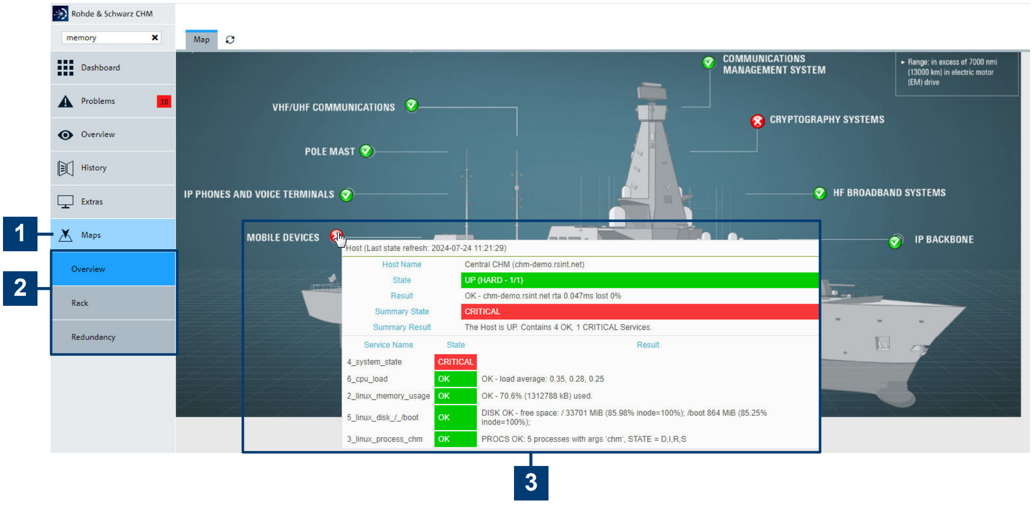
New, added or enhanced host configuration keys

- snmp_connection
  This key activates the SNMP upstream interface.
  Read more: Section 6.9, "Configuring the SNMP upstream interface", on page 76

2.3.2 R&S CHM v2402

New graphical system views (maps)

You can now configure graphical elements in R&S CHM. These elements let you visually track the system's status on fully customizable maps, providing a more intuitive and comprehensive understanding of the system's operation. On the web GUI, you can find all configured graphical system views under "Maps".



Host (Last state refresh: 2024-07-24 11:21:29)

Host Name	Central CHM (chm-demo.rsint.net)
State	UP (HARD - 1/1)
Result	OK - chm-demo.rsint.net rta 0.047ms lost 0%
Summary State	CRITICAL
Summary Result	The Host is UP. Contains 4 OK, 1 CRITICAL Services.

Service Name	State	Result
4_system_state	CRITICAL	
6_cpu_load	OK	OK - load average: 0.35, 0.28, 0.25
2_linux_memory_usage	OK	OK - 70.6% (1312788 kB) used.
5_linux_disk_/_/boot	OK	DISK OK - free space: / 33701 MiB (85.98% inode=100%); /boot 864 MiB (85.25% inode=100%);
3_linux_process_chm	OK	PROCS OK: 5 processes with args 'chm', STATE = D,I,R,S

Figure 2-2: Graphic system views (maps)

1 = "Maps" main menu
2 = Individual "Maps" views
3 = Mouse over on the status icon provides details. Select the status icon to navigate to the configured host or service.

Read more: Section 6.8, "Configuring graphical system views (maps)", on page 72

How to manage users in the local user database

You can list currently existing users, add users and delete users from the local user database.

Read more: "To manage users in the local user database" on page 57

How to check DoS settings and to monitor firewall rejects

If you are interested in the denial of service (DoS) settings and if you want to monitor firewall rejects, you can check the limits and enable firewall logging.

Read more: "To check DoS settings and to monitor firewall rejects" on page 31

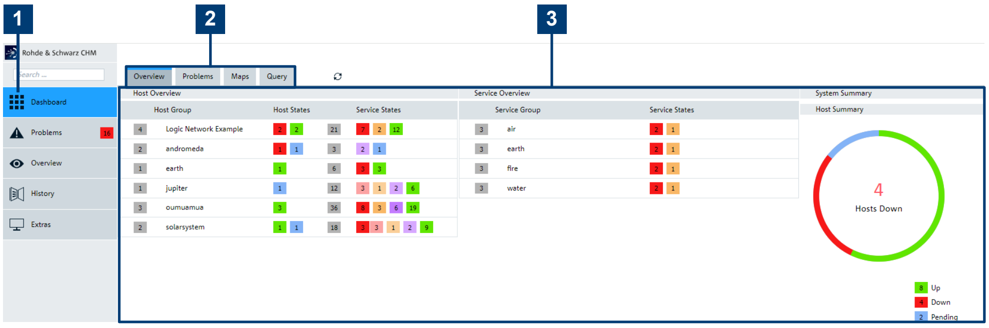
New system-wide location for Windows® client configurations

You can now create the `client_config.json` configuration file under a system-wide file location.

Read more: Section 4.3.3, "Connecting the client with the R&S CHM host",
on page 26

New or enhanced host configuration keys

- `dashboards`
  Configures the start page of the web GUI, the "Dashboard" (**1**). You can configure individual dashboard tabs (**2**) and the widgets (**3**) on these dashboards.



Host Overview

	Host Group	Host States		Service States
4	Logic Network Example	2 2	21	7 2 12
2	andromeda	1 1	3	2 1
1	earth	1	6	3 3
1	jupiter	1	12	3 1 2 6
3	oumuamua	3	36	8 3 6 19
2	solarsystem	1 1	18	3 3 1 2 9

Service Overview

	Service Group	Service States
3	air	2 1
3	earth	2 1
3	fire	2 1
3	water	2 1

System Summary

Host Summary

Figure 2-3: System-specific dashboard configuration

1 = "Dashboard" menu
2 = Multiple configured dashboard tabs
3 = Multiple configured widgets

Read more: dashboards on page 44

- maps
  The maps key lets you configure graphical system views (maps). On a background image, you can place status icons and labels. The maps key is used in the configuration on the top level and under specific hosts and status checks.
  Read more:
  – Graphical system view (maps) on page 74
  – maps on page 93
- notes
  The optional notes key lets you specify a multi-line text snippet that is shown on the web GUI for hosts and services.
  Read more: hosts on page 41 > notes.
- connections: [gb2pp]
  If you configure an R&S CHM host as a gb2pp server, you can configure a status check that queries this server for system or host group summary states.
  Read more: hosts on page 41

New or enhanced status checks

- tcp
  Checks if a TCP port is open and reachable from the R&S CHM host.
  Read more: tcp on page 138
- snmp_connection > hwinfo: true
  This optional key queries the SystemDescr OID and shows it on the web GUI > "Host" > "Result"
  Read more: snmp_connection on page 95
- gb2pp
  In combination with the host configuration connections: [gb2pp], the status check queries this server for system or host group summary states.

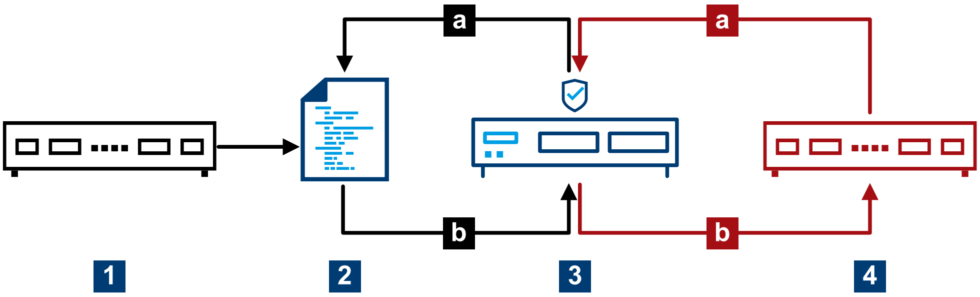


Figure 2-4: Conceptual representation of the gb2pp service check

1 = Host name: chmblack.example.net
2 = Monitoring data (gb2pp format)
3 = R&S TF5900M trusted filter IP
4 = Host name: chmred.example.net
a = Request monitoring data
b = Response

Read more: `gb2pp` on page 114

2.3.3 R&S CHM v2310

New feature R&S CHM client application

On Windows hosts, a status icon in the Windows notification area indicates the aggregated system status and lets you start the web GUI.

Read more: Section 4.3, "Installing R&S CHM clients", on page 25

New or enhanced status checks

- `nport`
  The optional `counter` key checks the port for frame, break, overrun and parity error counters.
  Read more: `nport` on page 126
- `system_state`
  Enables the Windows client interface and the check logic.
  Read more: `system_state` on page 137
- `connections: [client]`
  Use the `hosts` > `connections: [client]` key to configure a Windows host as a Windows client.
  Read more: `hosts` on page 41
- `navics`
  Monitors the status of NAVICS.
  You can redirect the result of the status check using a logic function as described in `logic` on page 48. For details, see the NAVICS example.
  Read more: `navics` on page 124

New or changed configuration keys

- logic
  The logic status check now provides the best function. In contrast to the worst function, the best function results in the best status result among different status results.
  Read more: logic on page 48
- health_host
  A new common key that lets you redirect a status request to a central monitoring host if you cannot obtain the status of the monitored component itself. You can use this key, e.g. in combination with the navics status check.
  Read more: health_host on page 92

2.3.4 R&S CHM v2306

New or enhanced status checks

- fortinet
  Monitors the status of a Fortinet controller.
  Read more: fortinet on page 113
- snmp_time
  Compare the time of a remote device with the time of the R&S CHM host by using SNMP.
  Read more: snmp_time on page 134
- snmp
  A new check that checks individual SNMP OIDs of a host for their return value.
  Read more: snmp on page 133
- trustedfilter
  New check for monitoring the status of the R&S TF5900M trusted filter IP firewall.
  Read more: trustedfilter on page 138
- snmp_connection
  New key that enhances SNMP configuration of R&S CHM hosts using SNMP.
  Read more: snmp_connection on page 95

Deprecated checks

Due to stability issues, do no longer use the bitdefender check.

2.3.5 R&S CHM v2302

Configuring distributed monitoring

Take advantage from extended monitoring configuration variants. Using the features for distributed monitoring, you can configure multiple R&S CHM instances that either monitor other hosts or devices, or that send monitoring results to R&S CHM hosts.

Thus, you can realize, e.g. a high-availability monitoring configuration or a multi-level configuration that is subdivided in several independent subsystems.

Read more: Section 6.10, "Configuring distributed monitoring", on page 78

New or changed configuration keys

These keys are new or changed under the host configuration (`hosts`):

- `checked by`
  A new key that lets you specify a dedicated R&S CHM host instance for host monitoring in the context of multi-level configurations.
- `connections`
  Value description enhanced.
- `displayname`
  A new key that lets you specify a user-friendly host name for display on the web GUI.
- `tags`
  The new `icinga2_ha` value lets you configure a secondary, high-availability R&S CHM host.
- `webinterface_url`
  In previous versions of the manual, the key was named `webinterface` by mistake.

Read more about the changes to `hosts`: `hosts` on page 41.

- `subsystems`
  New key in the context of multi-level configurations. Read more: `subsystems` on page 82
- `builtin`
  New key under `authentication`. If specified, it enables the built-in user database and thus provides a fallback login method to the web GUI if SSO, or LDAP is not available.

New or enhanced status checks

- `cputemp`
  New check for monitoring the average CPU temperature of Windows hosts.
  Read more: `cputemp` on page 106
- `file_content`
  Enhanced check that is now applicable to Windows agents.
  Read more: `file_content` on page 112
- `ping`
  Specify thresholds for **round-trip average time** and **package loss**.
  Read more: `ping` on page 132
- `tmr_radio`
  New check for TMR-MIB compatible radios. Read more: `tmr_radio` on page 138
- `os_service`
  A new check for monitoring the status of a Windows service.
  Read more: `os_service` on page 131

2.3.6 R&S CHM v2212

New single sign-on authentication options for web GUI users

R&S CHM now supports Kerberos-based single sign-on (SSO) authentication methods for web GUI users, see Section 6.5, "Configuring web GUI users", on page 57.

New configuration keys

Host configuration:

- `logic` on page 48

Checks:

- `passive` on page 131
- `dkn` on page 107

3 Introduction

The R&S CHM system status monitoring software provides an integrated, system-wide solution to collect status information continuously in a local area network (LAN). The software continuously performs checks for monitored hosts and services and evaluates the results. If R&S CHM detects an error condition, it creates an alert. The following figure provides an overview of a monitored system.

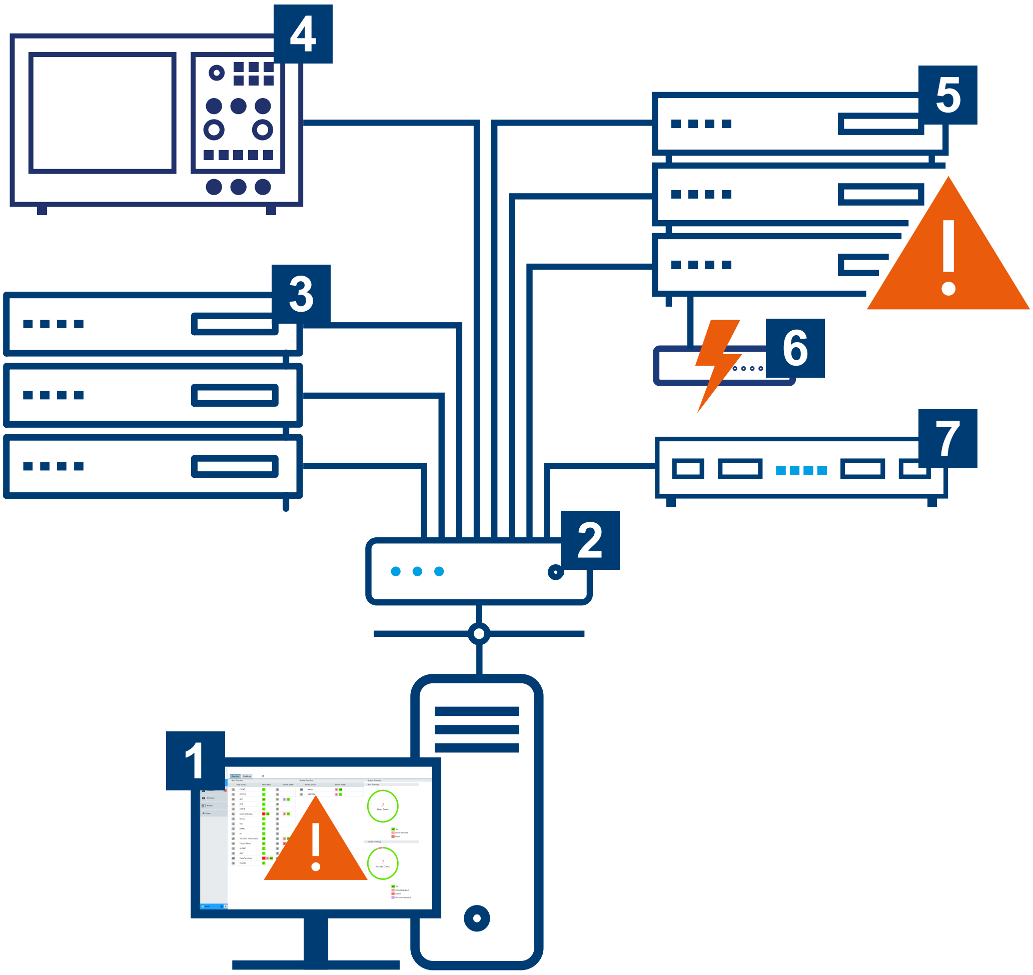


Figure 3-1: R&S CHM - status monitoring overview

1 = Computer with web-based user interface
2 = Network component (router, switch)
3 = Server hardware
4 = Rohde & Schwarz device
5 = Server hardware with error condition
6 = Uninterruptible power supply with error condition
7 = R&S CHM host that runs the status monitoring software

The R&S CHM software runs on a Linux server (**7**). You can access the web-based user interface on any standard computer in the network (**1**).

R&S CHM can fetch data from all connected and configured system components (**1** to **7**). Therefore, the operational state of the system is always under control. The downtime periods, due to maintenance operations or hardware failures, are reduced to a minimum.

Life of monitoring data

All monitoring data is retained for 90 days. Older data is purged from the database.

To monitor status information, system operators and administrators use the browser-based graphical user interface, in the following named as "web GUI".

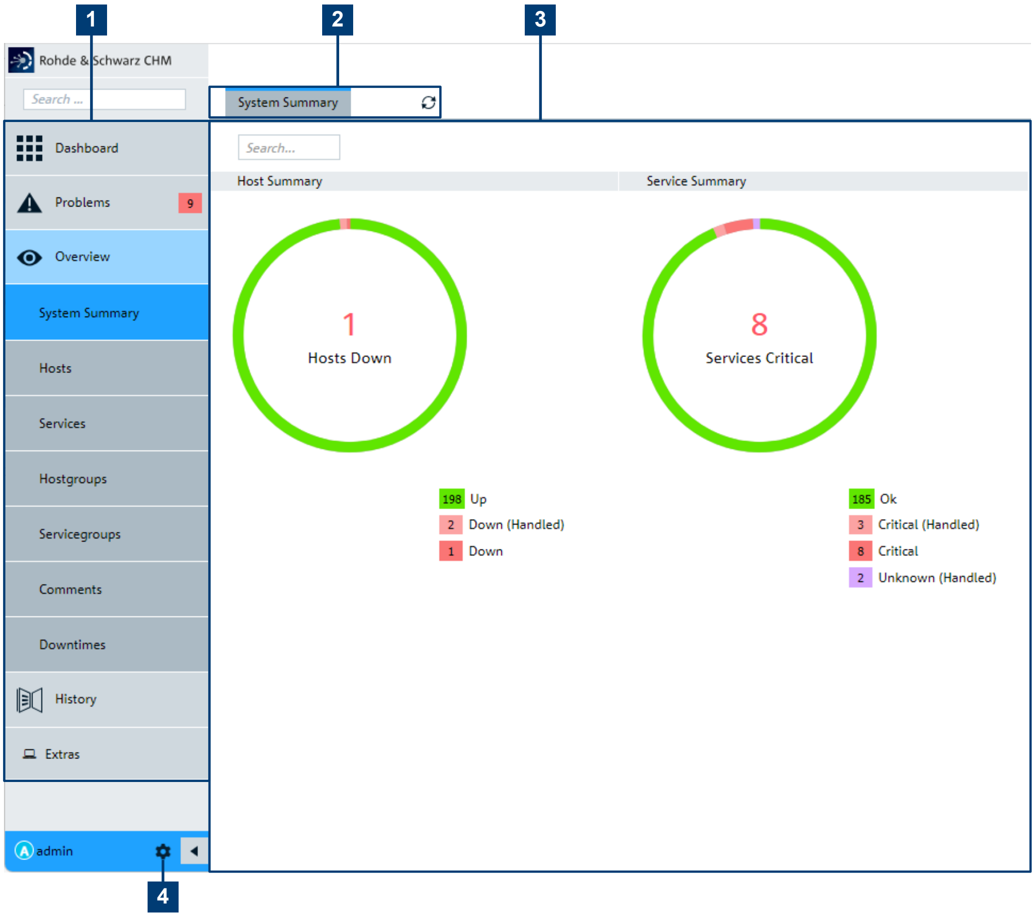


Figure 3-2: Web GUI for status monitoring

1 = Navigation and filter categories
2 = Additional filter categories
3 = Main area for status monitoring
4 = System-related pages and "Logout"

To configure the R&S CHM host from any computer in the LAN, system administrators can use an SSH client, such as PuTTY.

How to continue?

The next steps depend on your role as mentioned under "Target audience" on page 7.

- **Monitor system status information on the web GUI (operators and administrators)**
  Read the "R&S CHM System Status Monitoring" user manual.
- **Install and configure R&S CHM (system administrators, integrators)**
  These tasks address system administrators and software integrators:
  - Section 4, "Installing R&S CHM", on page 21
  - Section 6, "Configuring status monitoring", on page 37

4 Installing R&S CHM

Software installation is divided into these main parts:

- **R&S CHM host installation**
  The R&S CHM host software runs on Linux. Use the Rohde & Schwarz lifecycle software manager (LCSM) for installation. If LCSM is not available, follow the description in Section 4.1, "Installing the R&S CHM host without LCSM", on page 22.
- **R&S CHM agent installation**
  The agent software runs on monitored Windows- and Linux-based computers.
  – Section 4.2.1, "Installing Windows agents", on page 23
  – Section 4.2.2, "Installing Linux agents", on page 24

Before you start installation, review the minimum hardware and software requirements for the R&S CHM host and the agents.

Hardware and software requirements

You can install the R&S CHM host software on a server or a virtual machine (VM). Ensure that the R&S CHM host meets the minimum requirements listed in the following table. Keep in mind that the requirements increase with an increasing number of monitored system components and services.

Table 4-1: Requirements for R&S CHM hosts and Linux agents

Component	**Minimum requirements**
CPU	2 cores with 2 GHz
HDD	50 Gbyte
RAM	2 Gbyte
LAN adapter	1 Gbit/s, RJ-45 connector
Operating system	One of these Linux distributions: ● Oracle Linux v8.x or later ● CentOS Linux v7 (2009) Optional with hardening according to DISA standard.

Table 4-2: Requirements for Windows agents

Component	**Minimum requirements**
CPU	2 cores with 2 GHz
HDD	50 Gbyte
RAM	2 Gbyte
Operating system	Windows 10 build 1809 and later

System time requirements

All devices in the system that you monitor need to be time-synchronized using NTP.

4.1 Installing the R&S CHM host without LCSM

The R&S CHM host runs on specific Linux distributions. If you do not have a host running this operation system, we recommend downloading the Linux minimal version from the internet.

For comprehensive installation instructions, visit:

- Oracle Linux: https://docs.oracle.com/en/operating-systems/oracle-linux/8/install/
- CentOS Linux: https://docs.centos.org/en-US/centos/install-guide/

In the following procedure, only the main steps are provided.

To install Linux

1. Visit the homepage of the Linux distribution.
   - Oracle Linux: https://yum.oracle.com/oracle-linux-isos.html
   - CentOS Linux: https://www.centos.org/download/
2. Download an ISO image that suits the hardware architecture of your host, for example x86_64 for an Intel 64-bit server.
   - Oracle Linux v8.x (or later)
   - Linux v7 (2009)
3. Prepare the installation source.

   You can select from various options:
   - If you need a bootable physical media, prepare a DVD or a USB flash drive.
   - If you install Linux in a virtual machine, configure the virtual machine with at least the minimum requirements listed in Table 4-1. You can directly select the ISO image as the startup disk on your HDD.
   - If needed, you can also save the ISO image from a location on the network and boot it using NFS, FTP HTTP or HTTPS access methods.
4. Boot the installation media or ISO image.
5. Select "Install" in the boot menu and press [Enter].

   Anaconda, the Linux installer starts.
6. Follow the instructions on the screen.

   All installation options are properly configured, such as language, region, keyboard layout, date and time.
7. On the "INSTALLATION SUMMARY" screen, select "Begin Installation".

Installation of Linux starts.

Linux is installed on the host and ready for operation.

To install the R&S CHM host software

1. Ask your Rohde & Schwarz sales representative or applications engineer for providing the R&S CHM host software package.
2. Copy the `chm-<version>.tar.gz` archive to the R&S CHM host > `/root/`. For example, you can use WinSCP for this task.
3. Log in to the R&S CHM server, e.g. using SSH.
4. Change to the directory where the `chm-<version>.tar.gz` file resides.
5. Unpack the archive.

   ```
   # tar xfvz chm-*.tar.gz
   ```
6. Change to the extracted `chm` directory:

   ```
   # cd chm
   ```
7. Run the install script:

   ```
   # ./install-chm-server
   ```

   Installation takes a while. Wait until the `Completed` message is shown.

   The R&S CHM host is up and running.
   Continue with Section 6.3, "Changing the configuration", on page 40.

4.2 Installing R&S CHM agents

The agent is a program that runs remotely on a Windows or Linux computer. It helps provide information to the R&S CHM host. Contained PowerShell modules are signed on Windows and the Rohde & Schwarz certificate is installed.

Obtaining installers
Ask your Rohde & Schwarz sales representative or application engineer for providing the software package for R&S CHM Windows and Linux agents.

4.2.1 Installing Windows agents

R&S CHM supports the AllSigned execution policy.

1. Copy the `CHM_Windows_Agent_<version>.exe` installer to the Windows agent.
2. Run the `CHM_Windows_Agent_<version>.exe` installer.

3. If you install a Windows agent for gRPC-based R&S RAMON monitoring:
   a) Copy the `chmrd_<version>.msi` installer to the Windows agent.
   b) Run the `chmrd_<version>.msi` installer.

   The Windows agent is installed successfully.

   See also:
   - Section 5, "Deploying certificates", on page 32
   - Section 6.7, "Configuring R&S RAMON for monitoring", on page 68
   - Section 9.4, "Troubleshooting installation problems on Windows agents", on page 148

To configure a user-defined location for the package cache

During installation of Windows software, the installers add files to a location named package cache on your PC. If necessary, you can change the location for R&S CHM software installers on per-machine level using the Windows Registry Editor.

1. Select ⊞ "Start".
2. Type *registry editor*.
3. Select "Run as administrator".

   The Registry Editor opens.
4. Expand the "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\" key.
5. Create the following entries:
   a) Under "Policies", add the key "Wix".
   b) Under the "Wix" key, add the key "Burn".
      Resulting registry key:
      "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Wix\Burn"
   c) Under "Burn", add the string value "PackageCache".
   d) As value, enter the path without using environment variables. For example, `C:\my_package_cache_location\chm`

   R&S CHM software installers now use the user-defined location as the package cache location.

4.2.2 Installing Linux agents

1. Copy the `tar.gz` installer archive to the Linux agent.
2. Execute `# tar xfvz xxx.tar.gz`.
3. Execute `# ./install-chm-agent`

   The Linux agent is installed successfully.

4.3 Installing R&S CHM clients

An R&S CHM client (short: client) is an application to open the web GUI, including these additional features:

- Problem indication on a system tray icon .
- Autostart the application when logging on to the PC.
- Start the web GUI by using the tray icon. No need to install an additional browser.

To get a client up and running

Summary of necessary tasks:

1. Installing the client software
2. Extending the chm.yaml
3. Connecting the client with the R&S CHM host
4. Handling certificates
5. Starting the client for the first time
6. Configuring application logging
7. Setting up SSO

4.3.1 Installing the client software

R&S CHM supports the AllSigned execution policy.

1. Copy the `CHM_Client_<version>.msi` installer to a Windows agent or Windows PC.
2. Run the `CHM_Client_<version>.msi` installer.

   The client application is installed successfully.

4.3.2 Extending the chm.yaml

1. On the R&S CHM host, edit the `chm.yaml` file.
2. Enable the client interface and the check-logic.
   To do so, add the `system_state` key to the `checks` section of the R&S CHM host, here named `host1.de`.

```
hosts:
  - name: host1.de
    tags: [chm]
    checks:
      - ping:
      - system_state:
```

Figure 4-1: Code snippet - system_state check

See also: `system_state` on page 137

3. Specify the connection type for each client in the status monitoring system, e.g. the client named `host2.de`. The `connections: [client]` key ensures that the client can communicate with the R&S CHM host.

```
hosts:
  - name: host2.de
    connections: [client]
```

Figure 4-2: Code snippet - client connection type

Client interface, check logic and client connection type are configured properly.

4.3.3 Connecting the client with the R&S CHM host

In addition to the previous configuration in the chm.yaml file, you configure the connection between the client and the R&S CHM host in a JSON configuration file on the client.

1. Create the `client_config.json` text file in one of the following locations:
   - **User-specific**
     `%appdata%\chm-client\client_config.json`
     For example:
     `C:\Users\Administrator\AppData\Roaming\chm-client\client_config.json`
   - **System-wide**
     `%programdata%\chm-client\client_config.json`

2. Insert the following:

```
{
    "api": {
        "host": "<chm_host>"
    }
}
```

Figure 4-3: Client - JSON configuration file - minimal information

3. Substitute `<chm_host>` with the name of your R&S CHM host as specified in the `chm.yaml` file on the R&S CHM host. For example:

```
{
    "api": {
        "host": "host1.de"
    }
}
```

Figure 4-4: Client - JSON configuration file - example R&S CHM host name

4.3.4 Handling certificates

A client needs an SSL certificate for the secure communication with the R&S CHM host. Certificate usage depends on the implementation and certificate type.

Option 1: The client runs on a Windows agent

If the computer already runs the agent software, the same certificates are used by the client and found automatically.

See also: Section 5, "Deploying certificates", on page 32

Option 2: Using central PKI/central CA-signed certificates

If a central public key infrastructure (PKI) is used in your system and the certificates are generated and distributed via the central PKI, the client can use these certificates.

By default, the client checks for a valid certificate under `%appdata%\chm-client\` and in the certificate folder of the agent, see Section 5.2, "Using CA-signed certificates", on page 35.

The following certificates are necessary:

- `hostname.key`
- `hostname.crt`

If you want to use another certificate location, you can add this information to the JAML configuration file that you have created in Section 4.3.3, "Connecting the client with the R&S CHM host", on page 26.

Example:

JSON configuration file on the client. As a path separator, use double backslashes \\:

```
{
    "api": {
        "host": "host1.de",
        "client_crt": "C:\\temp\\someOtherCertificate.crt",
        "client_key": "C:\\temp\\someOtherCertificate.key"
    }
}
```

4.3.5 Starting the client for the first time

The following steps are only necessary if the client application is not installed on an agent and thus the HTTPS certificate is not installed in the certificate store yet.

1. Open the web GUI for the first time.
   a) If necessary, select Windows notification area > "Show hidden icons".
   b) Right-click > "Open".

   The client prompts you to import the HTTPS certificate.

2. Confirm the request to install the certificate.

3. On the log on page, enter your credentials.

   The client successfully connects to the R&S CHM host using the HTTPS protocol.

If you start the client without a valid configuration file, it automatically creates this file under `%appdata%\chm-client\`. Open this file and specify the right R&S CHM host name. See Section 4.3.3, "Connecting the client with the R&S CHM host", on page 26.

4.3.5.1 Configuring application logging

You can define to where a client logs to and the amount of logged information. To do so, specify an additional `"logging"` object in the JSON configuration file.

Example:

JSON configuration file on the client with optional `"logging"` settings.

```
{
    "api": {
        "host": "host1.de"
    },
    "logging": {
        "logger": "Console",
        "log_level": 3,
     }
}
```

Log types

- `"logger": "Console"`
  Recommended logging method. Logs everything on the console in which the client is started (default). If started using the `*.exe`, the logs go unnoticed.
- `"logger": "File"`
  Logs everything in a file, without file rotation. By default, the file is located here if you do not specify the file path:
  `%appdata%\chm-client\chm_client_log.txt`
  If you specify a different file location, use double backslashes (\\) as path separator:

`"file_path": "<drive>\\<folder>\\LogFileName.txt".`

Log level

The log level `"log_level": <number>` specifies the amount and type of logged information. The levels meet the Microsoft log level standard.

Table 4-3: Log level overview

Log level	Meaning/description	
0	Trace	Logs contain the most detailed messages (default). These messages can contain sensitive application data. These messages are disabled by default. Never enable them in a production environment.
1	Debug	Logs are used for interactive investigation during development. These logs primarily contain information useful for debugging and have no long-term value.
2	Information	Logs track the general flow of the application. These logs have long-term value.
3	Warning	Logs highlight an abnormal or unexpected event in the application flow, but do not otherwise cause the application execution to stop.
4	Error	Logs highlight when the current flow of execution is stopped due to a failure. These messages indicate a failure in the current activity, not an application-wide failure.
5	Critical	Logs describe an unrecoverable application or system crash, or a catastrophic failure that requires immediate attention.
6	None	Not used for writing log messages. Specifies that a logging category does not write any messages.

4.3.6 Setting up SSO

To allow the client to use single sign-on, specify an additional `"browser"` object in the JSON configuration file.

1. Open the `client_config.json` text file in one of the following locations:
   - **User-specific**
     `%appdata%\chm-client\client_config.json`
     For example:
     `C:\Users\Administrator\AppData\Roaming\chm-client\client_config.json`
   - **System-wide**
     `%programdata%\chm-client\client_config.json`
2. Insert the SSO configuration options as listed in Table 4-4. For an example, see Example "SSO browser configuration" on page 30.
3. If necessary, you can also add other configuration options as listed in the following table.

Table 4-4: SSO configuration options

Configuration	Explanation	Example value	Default value
auth_negotiate_delegate_whitelist	A comma-separated list of servers for which integrated authentication is enabled. Then any URL ending with the given servers is considered for integrated authentication. Without the * prefix, the URL must match exactly.	"auth_negotiate_delegate_whitelist": "*.example.com, *.foobar.com, *.baz"	NULL
auth_server_whitelist	A comma-separated list of servers for which delegation of user credentials is required. Without the * prefix, the URL has to match exactly.	"auth_server_whitelist": "*.example.com, *.foobar.com, *.baz"	NULL

Example: SSO browser configuration

Add the following lines to the `client_config.json` file. Substitute the server examples accordingly.

```
{
  "browser": {
    "auth_negotiate_delegate_whitelist": "*.example.com, *.foobar.com, *.baz"
    "auth_server_whitelist": "*.example.com, *.foobar.com, *.baz"
  }
}
```

You can configure additional browser settings as necessary.

Table 4-5: Additional browser settings

Configuration	Explanation	Example value	Default value
disable_renderer_backgrounding	Prevents Chrome from lowering the priority of invisible pages' renderer processes. This flag is global to all renderer processes, if you only want to disable throttling in one window, you can take the hack of playing silent audio.	"disable_renderer_backgrounding": "true"	False
enable_logging	Prints Chrome's logging to stderr or a log file.	"enable_logging": "true"	False
lang	Set a custom locale.	"lang": "en_US"	SystemDefault
ignore_certificate_errors	Ignores certificate-related errors. **Note**: This setting removes the check for certificate validation. This will reduce the overall system security and can lead to man-in-the-middle and similar attacks.	"ignore_certificate_errors": "true"	False

4.4 Firewall

The firewall rules are included in the software installer and thus set automatically. The following table informs about necessary connections.

Table 4-6: Firewall rules

Connection	Port	Protocol	Use case
CHM host → CHM host	4656	TCP	Transfer of system or hostgroup summary states between R&S CHM hosts over a trusted-filter device
CHM host → SNMP monitored device	161	SNMP	Collect monitoring information
Maintenance PC → CHM node	22	SSH	Optional SSH connection
PC → CHM node	80	HTTP	Viewing R&S CHM website in the browser (redirection to HTTPS)
PC → CHM node	443	HTTPS	Viewing R&S CHM website in the browser (encrypted connection)
CHM node → VMWare ESXi/vCenter	443	HTTPS	Monitoring of VMWare ESXi/vCenter status
Monitored item Windows/Linux → CHM node	5665	Icinga	Encrypted communication of Icinga (monitoring information)
Monitored item Windows/Linux ←→ CHM node	18005	Grpc	Encrypted communication of R&S CHM (monitoring and control information)

To check DoS settings and to monitor firewall rejects

In R&S CHM there is a preconfigured rate limiting for HTTPS and HTTP activated to reduce the probability of denial of service (DoS) attacks. The rate limiting is 50 packages per minute.

1. Verify the limit to check the `iptables` configuration.
   ```
   iptables -vL | grep https
   ```
2. Enable logging of the firewall.
   ```
   firewall-cmd --set-log-denied=all
   ```
3. Check the Linux journal for firewall rejects.
   ```
   journalctl -f | grep PROTO=TCP | grep REJECT | grep 443
   ```

5 Deploying certificates

Certificates protect the connections between the R&S CHM host and the R&S CHM agents. Without certificates, R&S CHM cannot monitor the system state of connected R&S CHM agents. Thus, we recommend deploying certificates on all R&S CHM agents.

The following figure serves as example system configuration. This configuration is used in the following procedures.

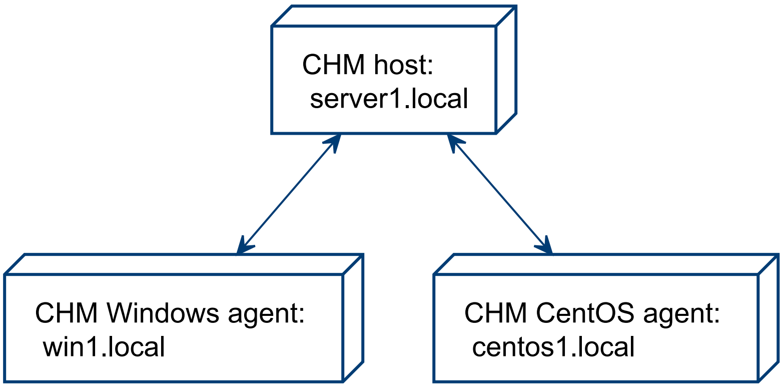


Figure 5-1: Example R&S CHM system

R&S CHM uses transport layer security (TLS) encryption to secure the communication between the R&S CHM host and the R&S CHM agents. By default, certificates are self-signed. Self-signed certificates are renewed automatically.

Also, you can use certificates that are provided by a central certificate authority (CA). If you want to use certificates from a central CA, contact your certificate manager. Self-signed certificates and a CA are generated automatically on the R&S CHM host during software installation.

Change all certificates before your system goes live.

5.1 Using self-singed certificates

You can use self-signed certificates as follows:

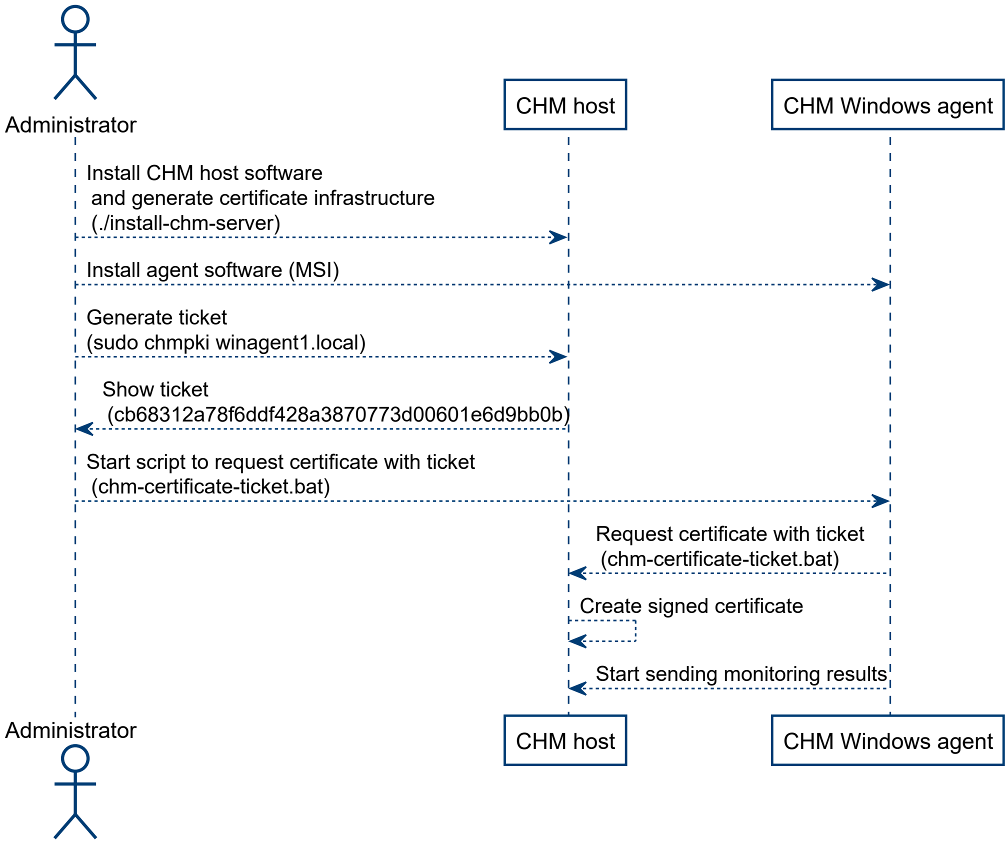
- With pregenerated tickets, see "To deploy certificates with tickets" on page 33.

- With certificate signing requests (CSR), see "To deploy certificates with signing request" on page 34.

As a prerequisite for creating certificates, the R&S CHM host must be installed and online.

To deploy certificates with tickets

The following figure shows the general workflow if you use self-signed certificates with tickets.


Administrator
CHM host
CHM Windows agent
Install CHM host software
and generate certificate infrastructure
(./install-chm-server)
Install agent software (MSI)
Generate ticket
(sudo chmpki winagent1.local)
Show ticket
(cb68312a78f6ddf428a3870773d00601e6d9bb0b)
Start script to request certificate with ticket
(chm-certificate-ticket.bat)
Request certificate with ticket
(chm-certificate-ticket.bat)
Create signed certificate
Start sending monitoring results
Administrator
CHM host
CHM Windows agent

1. Log in to the shell of server1.local using ssh.
2. Execute `sudo chmpki win1.local`.

   win1.local must be the FQDN of the windows agent.

   A generated ticket is shown.
3. Note down that ticket.
4. Connect to the windows host.
5. Create certificates:
   - **On Windows**, run this batch file as an administrator:
     `%programfiles%\chm\chm-certificate-ticket.bat`

- **On Linux**, issue this command:
  `chm-certificate-ticket`

The script prompts you for the server you want to connect to.

6. Enter `server1.local` and the ticket identifier.

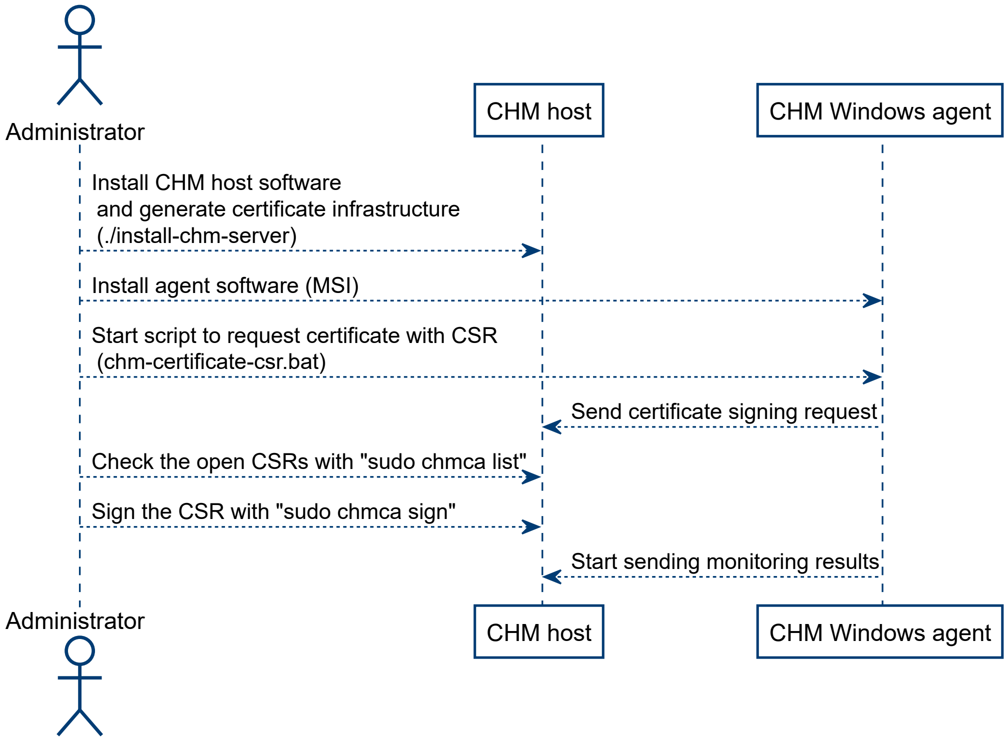
   The script creates the necessary certificates and configuration.

   If necessary, you can call the script with command-line arguments to execute it silently:

   - **On Windows**, run the batch file with parameters, all on one line:
     ```
     chm-certificate-ticket.bat
     server1.local cb68312a78f6ddf428a3870773d00601e6d9bb0b
     ```
   - **On Linux**, run this command with parameters, all on one line:
     ```
     chm-certificate-ticket
     server1.local cb68312a78f6ddf428a3870773d00601e6d9bb0a
     ```

To deploy certificates with signing request

The following figure shows the general workflow if you use self-signed certificates with certificate signing request (CSR).


Administrator
CHM host
CHM Windows agent
Install CHM host software
and generate certificate infrastructure
(./install-chm-server)
Install agent software (MSI)
Start script to request certificate with CSR
(chm-certificate-csr.bat)
Send certificate signing request
Check the open CSRs with "sudo chmca list"
Sign the CSR with "sudo chmca sign"
Start sending monitoring results
Administrator
CHM host
CHM Windows agent

1. Execute the configuration script:
   `%programfiles%\chm\chm-certificate-csr.bat`

   The script prompts you for the server you want to connect to.

2. Type in `server1.local`.

   The script requests the certificate at the R&S CHM host and generates it.

3. Log in to the server via ssh.
4. Execute `sudo chmca list`.

   All signing requests are shown, e.g.

```
Fingerprint                                                      | Timestamp           | Signed | Subject
-----------------------------------------------------------------|---------------------|--------|--------
403da5b228df384f07f980f45ba50202529cded7c8182abf96740660caa09727 | 2021/09/06 17:02:40 | *      | CN = win1.local
71700c28445109416dd7102038962ac3fd421fbb349a6e7303b6033ec1772850 | 2021/09/06 17:20:02 |        | CN = win2.local
```

5. Execute this command to approve the sign request from, e.g win1.local.

   `sudo chmca sign 403da5b228df384f07f980f45ba50202529cded7c8182abf96740660caa09727`

   Note: Ensure that the timestamp and CN are correct to ensure that only valid requests are singed.

   The Windows agent can send its monitoring results to the R&S CHM host.

5.2 Using CA-signed certificates

As an alternative to self-signed certificates, your company can use private certificate authorities to issue certificates for your internal servers.

We recommend using the following naming conventions:

- Certificate of the root CA: `ca.crt`
- Certificate of the server: `<fqdn>.crt`, where `fqdn` is the fully qualified domain name (FQDN).

1. Obtain the certificates from your certification authority.
2. Copy the certificates to these locations:

System component	Location
R&S CHM host	`/var/lib/icinga2/certs/`
Windows agent	`%programdata%\icinga2\var\lib\icinga2\certs\`
Linux agent	`/var/lib/icinga2/certs/`

5.3 Removing self-signed certificates

If necessary, you can remove all certificates on the R&S CHM host and the agents.

ⓘ If you remove the certificates, system status monitoring is no longer possible.

► Execute these commands:

- **On the R&S CHM host**: `sudo chm_clean_certificates`
- **On Windows agents**:
  `%programfiles%\chm\chm-clean_certificates.bat`
- **On Linux agents**: `sudo chm_clean_certificates`

5.4 Configuring a user-defined certificate location on Windows hosts

You can configure a common folder with all the needed certificates and make icinga use this folder instead of the default folder `C:\ProgramData\icinga2\var\lib\icinga2\certs\`. To reach this goal, we need a symbolic link that points to the common folder with the certificates.

To create the symbolic link

1. If certificates are already installed in the default folder, move them to the new location, i.e. the `Target` folder, e.g. `C:\Certificates`.
2. If existing, remove the (now empty) default folder:
   `C:\ProgramData\icinga2\var\lib\icinga2\certs`
3. You can use one of the following methods to create the symbolic link.
   - **In a Command Prompt window**
     Syntax:
     ```
     mklink /D Link Target
     ```
     Example:
     ```
     mklink /D "C:\ProgramData\icinga2\var\lib\icinga2\certs" "C:\Certificates"
     ```
   - **In the PowerShell**
     Syntax:
     ```
     New-Item -Path LINK -ItemType SymbolicLink -Value TARGET
     ```
     Example:
     ```
     New-Item -Path "C:\ProgramData\icinga2\var\lib\icinga2\certs" -ItemType SymbolicLink -Value "C:\Certificates"
     ```

6 Configuring status monitoring

Here, you can find all steps that are necessary to configure R&S CHM for system status monitoring. All data is contained in an editable configuration file. The configuration file is written in YAML v1.2 notation standard.

YAML is a human readable data serialization language for all programming languages. YAML is a case-sensitive language. It uses indentation with one or more spaces to represent the structure. Dashes (-) are used to represent the sequences (lists) and colons (:) are used to represent key-value pairs. The upper part of the configuration file on the R&S CHM host gives you an impression how this language looks like.

```
hosts:
  - name: host1.de
    displayname: CHM host
    tags: [chm]
    authentication:
      monitoring:
        - ldap:
            server: ldapserv.ourlocal.net
            port: 35636
            encryption: ldaps
            base_dn: ou=ldap_users,dc=ldapserv,dc=ourlocal,dc=net
            user_class: user
            user_name_attr: sAMAccountName
            bind_dn: service_user
            bind_pwd_path: ldap/service_user
    authorization:
[...]
```

Related information

- For more information about YAML, see Section 6.1, "Introduction to the YAML syntax", on page 38.
- For a YAML syntax reference, see the YAML website at https://yaml.org/refcard.html.

6.1 Introduction to the YAML syntax

The YAML syntax contains different kinds of data blocks:

- A **sequence** with values that are listed in a specific order. The sequence starts with a dash and a space, e.g. `- ping`.
- A simple **mapping** between key and value pairs. A key must be unique; the order does not matter.

A third type is called **scalar**, which is arbitrary data, such as strings, integers.

Data blocks can be written in block style or flow style.

Example: Sequence data blocks

A list of items in block style.

```
checks:
 - ping:
 - os_memory:
 - os_process:
```

A list of items in flow style.

```
host: [ping , os_memory , os_process]
```

Example: Mapping data blocks

```
snmp_connection:
  port: 161
  version: 2
  community: public
```

Example: Dictionary

This data block is a more complex collection of `key: value` pairs. Each pair can be nested with numerous options.

```
hosts:
  - domainname: chm-host.domain.net
    connections: [local]
    tags: [chm]
    hostgroups: [germany, bavaria]
    checks:
      - icinga2_cluster:
      - dhcp:
      - dns:
```

Table 6-1: Indicator characters - excerpt from the YAML syntax

Collection indicators	
:	Value indicator. In threshold configurations, the colon (:) indicates the edges of the interval, see also `thresholds` on page 98
-	Nested series entry indicator.

,	Separate in-line branch entries.
[ ]	Surround in-line series branch.
{ }	Surround in-line keyed branch.
Misc indicators	
#	Throwaway comment indicator.

Use single quotes (`' '`) in YAML if your string value includes special characters. For example, you possibly need single quotes around strings that contain these special characters:

`{, }, [, ], ,, &, :, *, #, ?, |, -, <, >, =, !, %, @, \.`

For details, see the YAML specification at https://yaml.org/spec in version v1.2.

6.2 Understanding aggregated states

Besides the individual status checks with their individual states, R&S CHM provides a state aggregation logic for the whole system and for host groups. The summarized states are accessible to different system components. Services that are acknowledged or in a downtime are excluded from state calculation. For more information about handling issues ("Acknowledge", "Schedule downtime"), see the "R&S CHM System Status Monitoring" user manual or help.

System state

The system state is defined as the worst state of all status checks within the system. The following figure shows the logic behind the system state and shows the components that use the system state. For example, the SNMP upstream interface, the gp2pp server check over an R&S trusted filter and the R&S CHM client can process the system state.

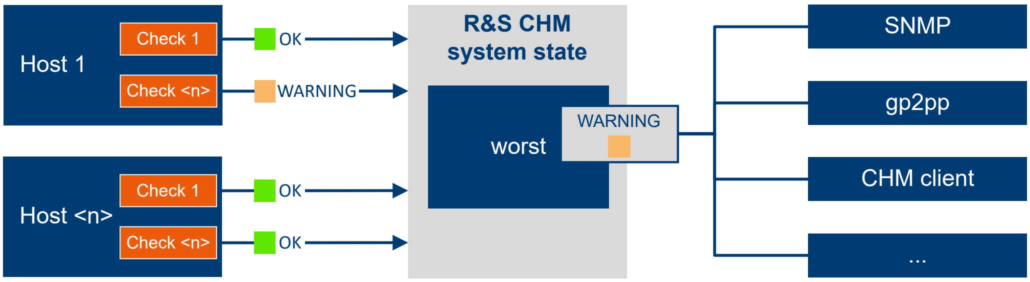


Figure 6-1: Aggregated state of hosts and checks

Host group state

Also, R&S CHM provides an aggregated host group state. The host group state is defined as the worst status of all status checks within that host group. For example, the

SNMP upstream interface and the gp2pp server check over an R&S trusted filter can process the host group state.

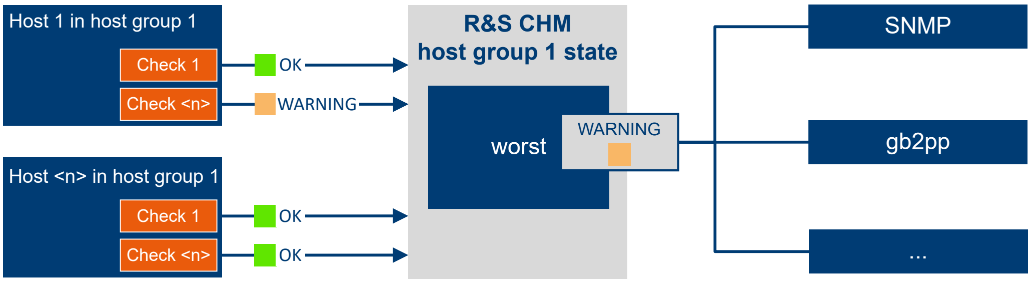


Figure 6-2: Aggregated state of hosts that belong to a host group

See also:

- Section 6.9, "Configuring the SNMP upstream interface", on page 76
- `gb2pp` on page 114

6.3 Changing the configuration

System administrators with *root* user account can configure R&S CHM and additional monitoring hosts and services. All configurations are defined in a single configuration file, which is the central configuration file for all objects that you want to monitor in the network.

To access the configuration file

► On the R&S CHM host, you can find the configuration file here:
`# /etc/opt/rohde-schwarz/chm/chm.yaml`

You can edit the file locally. Alternatively, you can transfer the configuration file to another PC, e.g. using WinSCP with SFTP or FTPS protocols. If finished, transfer it back to its original location on the R&S CHM host.

To edit the configuration file

1. Open the chm.yaml file in an editor.
   - On the local R&S CHM host, you can use the vi editor:
     `vi chm.yaml`
   - On a remote Windows host, you can use Windows Notepad or a more comfortable text editor with YAML syntax highlighting, e.g. Notepad++.
2. In the editor, navigate to the sequence item.
3. Add the key-value pairs.
4. Save the file.
5. If necessary, transfer the file back to its location on the R&S CHM host (`/etc/opt/rohde-schwarz/chm/chm.yaml`).

6. Restart these services on the R&S CHM host to take the changes effect:

   ```
   # sudo systemctl restart chm
   # sudo systemctl restart icinga2
   ```

   R&S CHM checks the syntax. If the syntax checks failed, edit the configuration file again and correct all syntax errors.

   If the syntax check was successful, the changes are applied. For example, you can monitor newly configured services on the web GUI.

Check if the services are running:

```
# sudo systemctl status chm
# sudo systemctl status icinga2
```

6.4 Configuring hosts

Here, you find detailed information on host configuration, including host-specific keys in the chm.yaml file.

A **host** is an independent device in the system. It is addressed and monitored by R&S CHM. For example, a host is a Windows PC, a Linux virtual machine or a device that you monitor using SNMP.

Hosts are characterized by several attributes, and several **checks** are subordinated to them. Each check verifies the host for an intended status, e.g. available disk space, temperature or other hardware status.

hosts (Hosts)

The hosts dictionary consists of a list of all host elements for the whole system to be monitored.

Here, you specify the configuration and the checks for all hosts where R&S CHM is installed or that are monitored by R&S CHM.

Parameters:

name

string

Name of the host, i.e. the name of the R&S CHM host, a R&S CHM agent or an SNMP device that is monitored. A host instance always starts with the `name` key.
The first host instance in the file always denotes an **R&S CHM host**.

displayname

string

Shows this name on the web GUI instead of the specified `name` (optional).

dashboards

Configures the contents of the "Dashboard". See `dashboards` on page 44.

notes

string

Specifies a text snippet for hosts and services (optional). You can add a detailed description of the location or details on how to handle errors. Use the `<br>` tag to write a multi-line note.
R&S CHM shows this text snippet on the web GUI > "Service"/"Host" tab > "Problem handling".

tags

[chm] | [icinga2_ha]

Assigns a role to a host in the status monitoring system.

[chm]
Assigns the role "master" to an exclusive R&S CHM host or the role "primary master" to one of the R&S CHM hosts for a high-availability status monitoring configuration. For more information about the roles, see Section 6.10, "Configuring distributed monitoring", on page 78.
An R&S CHM host that is tagged with `[chm]` starts a monitoring system in which all hosts are synchronized regarding monitoring state. All hosts that are specified beneath are part of this monitoring system.
The next R&S CHM host instance tagged with `[chm]` starts the next monitoring system, and so forth. In combination with `exports`, you can configure multiple monitoring systems. These hosts are not synchronized, because the R&S CHM hosts are separated from each other, e.g. by a security gateway.

[icinga2_ha]
Assigns the role "secondary master" to a second R&S CHM host in a high-availability status monitoring system. All hosts that are tagged `[icinga2_ha]` belong to the same overall status monitoring system as the associated "primary master". Both R&S CHM hosts, primary master and secondary master, are fully synchronized regarding monitoring state.
For usage scenarios, see Section 6.10.1, "Configuring high availability monitoring", on page 79 and Section 6.10.4, "Configuring multi-level HA monitoring", on page 88.

logic

Combines status values from multiple checks to a single, aggregated status value. See `logic` on page 48.

logging

Configures the severity and the facility for event logging on the R&S CHM host. See `logging` on page 53.

exports

Configures an R&S CHM host so that it sends status monitoring information to another R&S CHM host (optional). See `exports` on page 47.

authentication

Configures LDAP-based user authentication. See `authentication` on page 59.

authorization

Configures user authorization. See `authorization` on page 63.

webinterface_url

string

Configures a hyperlink to the management web interface of the host. See `webinterface_url` on page 56.

connections

[local] | icinga2_win] | [icinga2_linux] | [snmp] | [client] | [gb2pp]

Defines how R&S CHM communicates with this host.

[local]
If you configure an operating system-dependent check on a master, specify `[local]`. This setting ensures that the right check plugin is used for all checks that depend on the operating system (Windows or Linux). For example, `load` is such a check.

[icinga2_win]
Denotes Windows agent. The checks run on this agent. The agent sends the check results to the master.

[icinga2_linux]
Denotes a Linux agent. The checks run on this agent. The agent sends the check results to the master.
Check plugin for Linux agents and satellites in multi-level monitoring configurations.

[snmp]
Denotes a host that is monitored by the R&S CHM host by using SNMP. The host is not installed on an R&S CHM agent. All checks are performed on the R&S CHM host and the check results are obtained by active checks.

[client]
Denotes an R&S CHM client. Specify `[client]` for a host that runs the R&S CHM client application.
How to: Section 4.3, "Installing R&S CHM clients", on page 25

[gb2pp]
Denotes that this R&S CHM host is configured as a gb2pp server.
For the necessary checks and for examples, see `gb2pp` on page 114.

checked_by

string

Specifies the R&S CHM instance that monitors this host (optional). You can specify this key for all hosts that are not a master, a satellite or an agent. For an example, see Example "YAML configuration: multi-level HA monitoring" on page 90. Without this key, the default applies. Hence, hosts are monitored by default by the R&S CHM instance that is located in the same subsystem and that is not configured as a high availability host.

hostgroups

[comma-separated list of strings]

List of groups the host belongs to. The groups help identify the host on the web GUI.

snmp_connection

Activates the SNMP upstream interface (optional).
How to: Section 6.9, "Configuring the SNMP upstream interface", on page 76

checks

Parent key for all status checks. The first check is always a host check that checks if the host is reachable. All following checks are service checks. These checks provide information about the health states of the checked resources. See Section 7, "Configuring status checks", on page 99.

For detailed R&S CHM host configuration examples, see Section 8.1, "R&S CHM host configuration", on page 143 and Section 8.2, "Linux host configurations", on page 144.

Example:

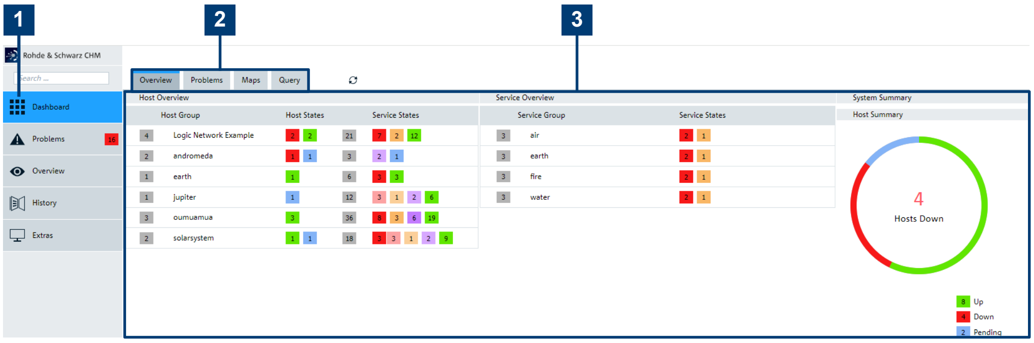
Single R&S CHM host configuration with some high-level keys.

```
hosts:
  - name: host1.de
    displayname: CHM master
    tags: [chm]
    logic:
      aggregation1:
        function: worst
        ins: [input1, input2, input 3]
    logging:
      severity: info
      facility: local0
    authentication:
    authorization:
    webinterface:
    connections: [icinga2_linux]
    hostgroups: [monitoring, control]
    checks:      # The checks for this host
```

dashboards (Main elements on the web GUI > "Dashboard")

Configures the start page of the web GUI, the "Dashboard" (**1**). You can configure individual dashboard tabs (**2**) and the widgets (**3**) on these dashboards.

The `dashboards` key is optional. If it is not specified, the web GUI shows the default dashboards "Overview" and "Problems".



	Host Group	Host States		Service States
4	Logic Network Example	2 2	21	7 2 12
2	andromeda	1 1	3	2 1
1	earth	1	6	3 3
1	jupiter	1	12	3 1 2 6
3	oumuamua	3	36	8 3 6 19
2	solarsystem	1 1	18	3 3 1 2 9

	Service Group	Service States
3	air	2 1
3	earth	2 1
3	fire	2 1
3	water	2 1

Figure 6-3: System-specific dashboard configuration

1 = "Dashboard" menu
2 = Multiple configured dashboard tabs
3 = Multiple configured widgets

Related parameters

- `Graphical system view (maps)` on page 74
- `widgets` on page 46

Parameters:

name
string
Label of a dashboard tab (**2**). The `name` must not contain dots (`.`).

widgets
The areas on an individual dashboard (**3**). For details, see `widgets` on page 46.

Example:

```
hosts:
  # First host list entry. (1)
  - name: host1.de
    displayname: CHM host
    dashboards:
      - name: Maps
        widgets:
          - name: Overview Map
            content: "Map: Overview1"
          - name: A custom query
            content: "Query: filter_pattern" # (2)
          - content: Host Problems - unhandled
      - name: Overview
        widgets:
          - content: Service Problems - unhandled
          - name: Overwrite the name with a custom name
            content: Host Problems - unhandled
```

(1) By convention the first hosts entry is the R&S CHM host.

(2) In the above example, the *`filter_pattern`* looks like this:

```
monitoring/list/hosts?hostgroup_name=andromeda&sort=host_severity
```

widgets (Areas on a dashboard)

Configures the areas of an individual dashboard. An individual widget consists of a name and content.

Related parameters

- dashboards on page 44
- Graphical system view (maps) on page 74

Parameters:

name
string
Heading of an area on a dashboard tab (optional for `content: <preset_value>`). If you configure the `name` in combination with `content: <preset_value>`, this configuration results in a user-defined heading that overwrites the default heading.
The `name` is mandatory in combination with `content: "Map: <map_name>"` and `content: "Query: <pattern>" Query`).
The `name` must not contain dots (`.`).

content
Host Overview | Service Overview | Status View | System Summary | Recent Events | Host Problems - unhandled | Service Problems - unhandled | "Map: <map_name>" | "Query: <pattern>"
Contents of the widget.
Map: <map_name>
Name of the map as specified in Graphical system view (maps) on page 74 > `name`. Due to the colon (`:`), enclose the whole string in quotation marks.
Query: <filter_pattern>
Query pattern. Due to the colon (`:`), enclose the whole string in quotation marks.

Example:
Filter for a host name "Master" in the "Hosts" view:

```
monitoring/list/hosts?host=%2AMaster%2A
```

`%2A` is the HTML code for the asterisk (*) as a universal placeholder in a filter pattern.

Example:
Filter for a host group name "andromeda" in "Hostgroups" view:

```
monitoring/list/hosts?hostgroup_name=%2Aandromeda%2A
```

To create queries with complex filter patterns is reserved for R&S CHM experts. For suitable filter patterns, ask your Rohde & Schwarz support engineer.

For an example in the `dashboards` context, see dashboards on page 44.

exports (Export of status information)

If two R&S CHM systems are separated by a security gateway, you can configure this key to send status information from one R&S CHM host to the other R&S CHM host.

To do so, you configure the target R&S CHM host and the data format that is used by R&S CHM for sending status monitoring information.

The following figure explains the basic principles. R&S CHM sends status information from a **Domain B** to a separated **Domain A**. On its way, the status information is filtered by a security gateway. R&S CHM host (A) can monitor its own items and display the monitored items from R&S CHM host (B).

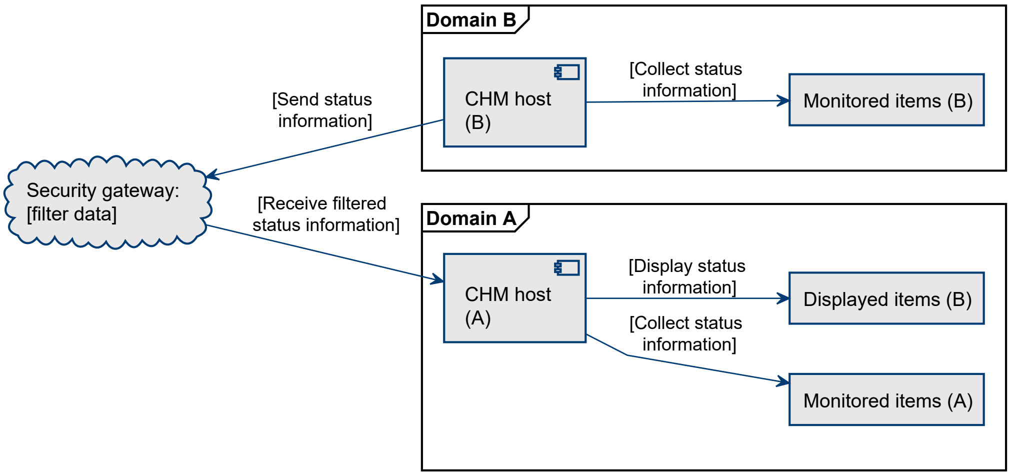

Domain B
CHM host (B)
[Collect status information]
Monitored items (B)
[Send status information]
Security gateway: [filter data]
[Receive filtered status information]
Domain A
CHM host (A)
[Display status information]
Displayed items (B)
[Collect status information]
Monitored items (A)

Figure 6-4: Exporting status information form domain B to domain A

Prerequisite

Both R&S CHM hosts need identical `chm.yaml` files. So, first change the file on one host. Then, transfer the file to the other host, e.g. using SSH. Example **2** at the end of this description shows the high-level structure of the `chm.yaml` file.

Parameters:

xmlhttp	Interface used for sending status information. This interface uses HTTP with content type application/xml on TCP port 5669.
target	Name of the R&S CHM host that receives the status information.
proxy	If the gateway acts as HTTP proxy, IP address or host name (optional).

Example: Configuration with two R&S CHM hosts and some high-level keys, including `exports` configuration.

```
hosts:
# First R&S CHM host
  - name: chm-k130-domain-A
    tags: [chm]
    connections: [local]
    logging:
      severity: debug
      facility: local0
    checks:
      - load:
      - os_process:
          name: icinga2
# Second R&S CHM host
  - name: chm-k130-domain-B
    tags: [chm]
    connections: [local]
    logging:
      severity: debug
      facility: local0
    exports:
      - xmlhttp:
          target: chm-k130-domain-A
          proxy: 1.2.3.4:5669
    checks:
      - load:
      - os_process:
          name: icinga2
```

logic (Combine logic status values)

R&S CHM system status monitoring lets you combine status values from multiple checks to a single, aggregated status value.

The following figure visualizes an example using the `worst` function. We assume that you monitor three components of a device, and DKN device 2 is defective. The `worst` function now takes the most critical value and hands it over to, e.g. the web GUI. The overall status indication that you configure using the `logic` key is 🟧 "WARNING".

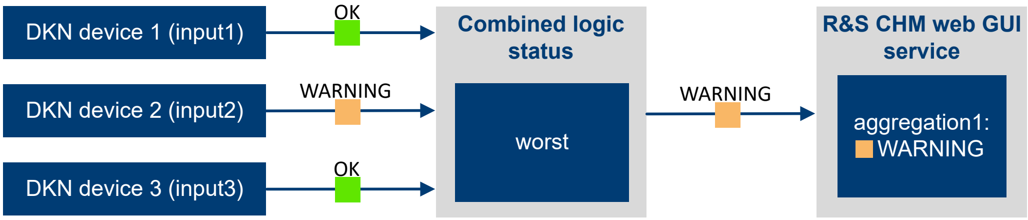
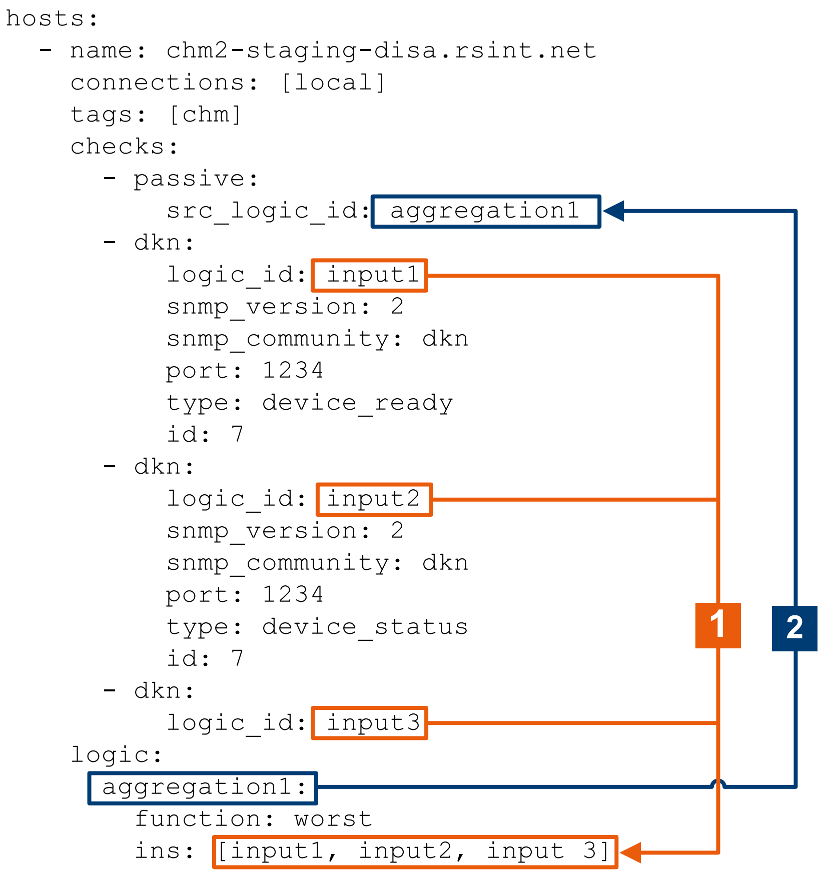


Figure 6-5: Combined logic status - "worst" example

The following figure adopts the previous example and explains how the specified keys are used to configure the logic function and to promote the aggregated status to the web GUI.



```
hosts:
  - name: chm2-staging-disa.rsint.net
    connections: [local]
    tags: [chm]
    checks:
      - passive:
          src_logic_id: aggregation1
      - dkn:
          logic_id: input1
          snmp_version: 2
          snmp_community: dkn
          port: 1234
          type: device_ready
          id: 7
      - dkn:
          logic_id: input2
          snmp_version: 2
          snmp_community: dkn
          port: 1234
          type: device_status
          id: 7
      - dkn:
          logic_id: input3
    logic:
     aggregation1:
        function: worst
        ins: [input1, input2, input 3]
```

Figure 6-6: Usage of involved keys

1 = Check with configured logic function instance
2 = Logic function instance used to promote the aggregated state to the web GUI

In detail, the previous configuration reads as follows: A logic identifier is assigned to each of the devices (`input1`, `input2`, `input3`). For logic function instance `aggregation1`, the logic function `worst` combines all input monitoring states and determines the most severe status as the *check result*. The `passive` key adopts the *check result* and shows it on the web GUI.

The next figure visualizes an example using the `best` function. We assume that you monitor two hosts and **host 2** (demodevice2.example.net) is defective. The `best` function now takes the best value and hands it over to the web GUI. The overall status indication that you configure using the `logic` key is 🟩 "UP".

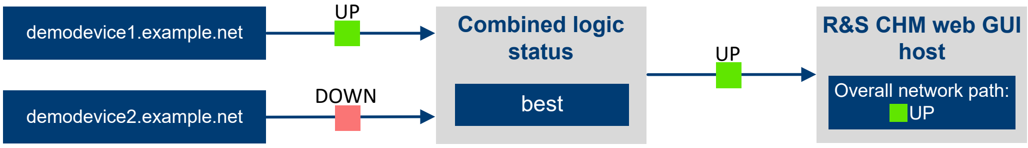


Figure 6-7: Combined logic status - "best" example

For a "best" coding example, see example "2a) Example (best function)" at the end of this section.

Parameters:

<log_funct_inst> string

Instance of a logic function. You can specify multiple instances that you configure using the following `function` key.
You can also specify such a logic function instance for a check using `logic_id` on page 93.

function worst | best

The logic that is used for aggregation of status values.

worst
From all checked status values, the most severe status value determines the aggregated status value.

best
From all checked status values, the best status value determines the aggregated status value.

ins [<logic_function_instance 1>, <logic_function_instance 2>, ..., <logic_function_instance n>]

List of input values that are evaluated by the logic function.
See also: `logic_id` on page 93.
You can list logic function instances that are specified here, under `logic`. Also, you can list logic function instances that you have specified for a dedicated check using the `logic_id` on page 93 key.

Example:

1) DKN example (worst function)

This example is copy ready. It is identical to the example in Figure 6-6.

```
hosts:
  - name: chm2-staging-disa.rsint.net
    connections: [local]
    tags: [chm]
    checks:
      - passive:
          src_logic_id: aggregation1
      - dkn:
          logic_id: input1
          snmp_connection:
            version: 2
            community: dkn
            port: 1234
          type: device_ready
          id: 7
      - dkn:
          logic_id: input2
          snmp_connection:
            version: 2
            community: dkn
            port: 1234
          type: device_status
          id: 7
      - dkn:
          logic_id: input3
    logic:
      aggregation1:
        function: worst
        ins: [input1, input2, input 3]
```

Example:

2a) Example (best function)

```
- name: chm-server.example.net
  displayname: "CHM Server"
  tags: [chm]
  ...
  logic:
    logic_path_available:
      function: best
      ins: [device1, device2]

...

- name: Overall network path
  checks:
    - passive:
        src_logic_id: logic_path_available

- name: demodevice1.example.net
  checks:
    - ping:
        logic_id: device1

- name: demodevice2.example.net
  checks:
    - ping:
        logic_id: device2
```

Example: **3) NAVICS example (worst function)**

This example uses the result of the `navics` status check also as host result. The result is redirected using a logic function.

```
hosts:
  - name: chm-demo.rsint.net
    displayname: "Central CHM"
    connections: [icinga2_api, local]
    tags: [chm]
    checks:
      - ping:
    logic:
      copy_of_navics_VT1:
        function: worst
        ins: [navics_VT1]                        # (1)

    - name: VT1.navics
        connections: [snmp]
        checks:
          - passive:
              src_logic_id: copy_of_navics_VT1   # (2)
          - navics:
              logic_id: navics_VT1               # (3)
              health_host: navics_server.local
              type: cwp
              eqid: VT1


    - name: navics_server.local
        connections: [snmp]
        snmp_connection:
          community: public
        checks:
          - ping:
```

(1) defines the logic function.
(2) receives the result from the logic function as host check result.
(3) sends the result to the logic function.

See also:

`passive` on page 131

`logic_id` on page 93

logging (System logging)

R&S CHM components send their log events into the Linux journal of the R&S CHM host. The journal is a binary, ring-buffer like database.

By default, Linux keeps the journal in volatile memory. You can persist messages to text files by using the `syslog` service. It reads the journal and exports to text files by some filter rules.

Note: Currently, R&S CHM does not provide means to change these export settings. If you use the `syslog` service, R&S CHM logging can cause high IO and CPU load and can degrade flash memory (SSDs). Ensure that only a subset of messages is exported, e.g. warning and higher.

For possible Linux logging options, see the related man pages.

Logging configuration

You configure the logging level in the `chm.yaml` file under the hosts key, see `hosts` on page 41.

Viewing logs

You can view the logs using the `# sudo journalctl` command.

Log events can originate at different components. For identification of the component, see Table 6-2.

Parameters:

severity

emerg | alert | crit | err | warning | notice | info | debug

Specifies the severity level, i.e. the importance of the message. For severity details, see Table 6-3.
If you change the severity, e.g. to `err`, only messages with severity `err` or higher are logged (`crit`, `alert`, `emerg`). For normal operation, we recommend severity `info`.

*RST: info

facility

local0 | local1 | local2 | local3 | local4 | local5 | local6 | local7

Specifies the type of system that is logging the message according to RFC 5424. Messages with different facilities can be handled differently.

local0
Locally used facility code. All R&S CHM components send their logs as facility `local0`.

*RST: local0

Example:

Logging configuration under the `hosts` key. The severities with numerical code "0" to "5" are logged:

```
logging:
  severity: notice
  facility: local0
```

Example:

Query of a specific component with `# journalctl -t <Identity>` or `# journalctl SYSLOG_IDENTITY=<Identity>`:
For example, query the monitoring web UI and the web server status.

```
# journalctl -t chm-monitoring-webui -t chm-httpd
```

Example: **Query of successful login, logout and failed login at the web interface**:

```
# journalctl | grep "User logged in"

# journalctl | grep "User logged out"

# journalctl | grep "User failed to authenticate"
```

Example: **Filter for certain facilities** using `journalctl SYSLOG_FACILITY=<facility_code>`:

```
# journalctl SYSLOG_FACILITY=16
```

For a list of facility codes and their meaning, see RFC5424.

Example: **Filter messages by severity** with `journalctl -p <severity_or_severity_rage>` or `journalctl PRIORITY=<numerical code>`:

```
# journalctl PRIORITY=6
```

Example: **Message output**:

```
Oct 07 11:25:48 test.local chm-httpd[10476]:
Thu Oct 07 11:25:48.797427 2021] [ssl:info]
pid 121267] [client 172.27.18.70:56854]
AH01964: Connection to child 2 established
(server test.local.net:443)
```

Table 6-2: Functional components

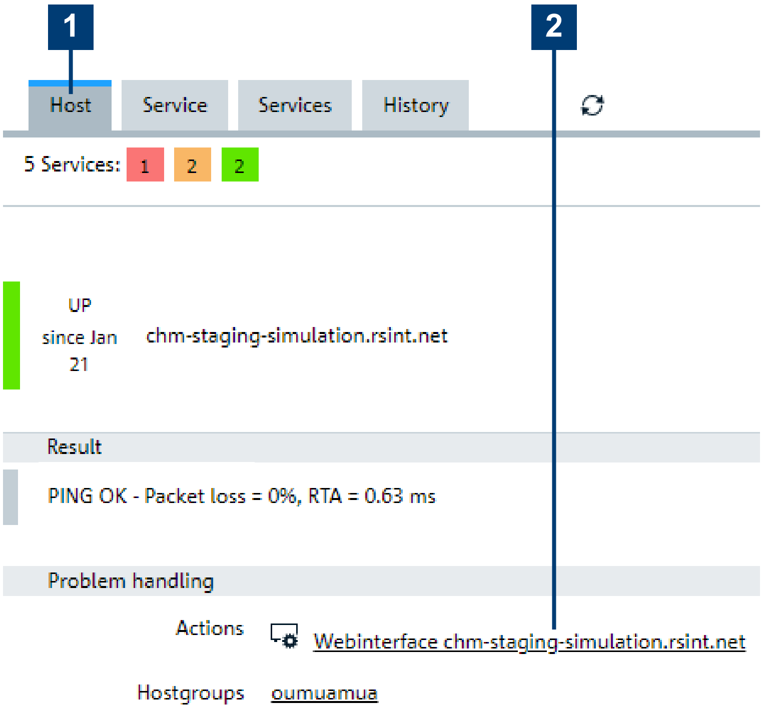
Component identity	Description
icinga2	Monitoring core
chm-monitoring-webui	Monitoring web UI
chm-monitoring-webui-audit	User login events at the monitoring web UI
chm-httpd	Web server status
chm-httpd-req	Web server request and responses

Table 6-3: Logging levels (severities) in order of decreasing importance

Parameter value	Numerical code	Description
emerg	0	Emergency - the system is unusable
alert	1	Alert - an action must be taken immediately
crit	2	Critical conditions
err	3	Error conditions
warning	4	Warning conditions
notice	5	Normal, but significant, condition
info	6	Informational message
debug	7	Debug-level message

webinterface_url (Hyperlink to management web interface)

Configure a hyperlink to the management web interface of the monitored host. R&S CHM shows the hyperlink on the web GUI.



chm-staging-simulation.rsint.net

PING OK - Packet loss = 0%, RTA = 0.63 ms

Webinterface chm-staging-simulation.rsint.net

Figure 6-8: Hyperlink to a web interface

1 = "Hosts" tab
2 = Hyperlink to the web interface of the host

Configuration details

Select from the following options:

- Compose the link automatically from the host name. This mechanism requires that the host name is specified as a fully qualified domain name. R&S CHM system status monitoring automatically adds `https://` in front of the host name to compose the hyperlink, e.g. `https://chm-staging-simulation.rsint.net`.
- Specify a dedicated URL, e.g. `https://rohde-schwarz.com`. The web GUI shows this hyperlink.
- Omit the parameter from the configuration to omit the entry on the web GUI.

HTTP or HTTPS web address of the web interface of the host. If the name of the host is configured as a URI, CHM automatically composes the hyperlink, e.g. `https://chm-staging-simulation.rsint.net`.

Example: Automatically compose hyperlink:

```
- name: chm-staging-simulation.rsint.net
  webinterface:
```

Resulting hyperlink:

```
https://chm-staging-simulation.rsint.net
```

Example: Specific hyperlink:

```
- name: chm-staging-simulation.rsint.net
  webinterface: https://rohde-schwarz.com
```

6.5 Configuring web GUI users

You can select from the following configuration methods to access the web GUI:

- Default local user database, see "Local user database (method 0)" on page 57.
- LDAP-based or Kerberos-based authentication method, without or with fallback to the local user database, see "LDAP and single sign-on authentication methods" on page 58.

Local user database (method 0)

This method is the default log in method. It works without external dependencies to an authentication server like Active Directory.

You can use the local web GUI users "admin" and "operator" if the following applies:

- Authentication is not configured in the `chm.yaml` file.
- The `builtin` key is configured as an authentication method.

Both local users and their permissions are predefined. The "admin" user gets all permissions (acknowledge, check, comment, downtime, monitoring, maps, checkdetails, manual). The "operator" user only gets the monitoring and the maps permissions.

For information on permissions details, see `authorization` on page 63 > `permissions`.

To manage users in the local user database

You can list, add and delete users from the local user database.

► Use the following commands:
  - `# chm_userlist`: Lists all currently existing users.
  - `# chm_useradd`: Adds a new user with password. The password is hidden on the command line while typing.
  - `# chm_userdel`: Deletes a user.

To change the passwords of existing users

1. Delete the user.

```
# chm_userdel
```

2. Add the user again with the new password. Use a unique and strong password that complies with the security policies in your company.
   `# chm_useradd`
3. Configure the user permissions for the newly added user in the `chm.yaml` file. See the following procedure ("To control the permissions of local web GUI users" on page 58.

Example:

The following examples show how to use the commands for user management.

List all users	Add a new user with password	To delete a user
# chm_userlist name ---------- admin operator test1 test2 ee uh (6 rows)	# chm_useradd new user: testuser new password: INSERT 0 1 completed	# chm_userdel user to remove: testuser DELETE 1 completed

To control the permissions of local web GUI users

You can use the local users "admin" and "operator" without further configuration. However, you can assign specific permissions to them, e.g. to the "operator".

1. Under the first `hosts` list entry, add the `authorization` key.
2. Configure the `permissions` for specific `roles`. For example, add `check` and `acknowledge` for `operators`.
   For all permissions and configuration details, see `authorization` on page 63 > `permissions`.

   You have configured specific permissions of the local web GUI user.

In a distributed system with several R&S CHM hosts, the local user database is not synchronized between the instances.

LDAP and single sign-on authentication methods

You can select between LDAP-based user authentication or Kerberos-based single sign-on (SSO) user authentication methods. R&S CHM then uses the configured method to restrict permissions and users that can access the web GUI. By using one of these methods, you can manage users or user groups centrally and enhance security.

Usage of all SSO methods requires several 3rd-party services, e.g. LDAP and Kerberos. Implementation and configuration of these services are not covered by this user guide. The services also require configuration of the web GUI users in the central user management of your organization. Ask the local system administrator for support.

Here, we describe the configuration of the `chm.yaml` and the requirements for using 3rd-party services.

R&S CHM supports user and group management with these directory services to log in to the web GUI:

- **LDAP with dedicated bind user (method 1)**
  LDAP bind uses the credentials of dedicated bind-users defined in the password store. Users log in with a password as stored in a central LDAP-service.
- **LDAP anonymous bind (method 2)**
  LDAP bind does not require credentials. Users log in with a password as stored in a central LDAP-service.
- **Kerberos SSO with SSSD for group information (method 3)**
  Users are logged in automatically with a ticket that they receive and cache during Windows- or Linux desktop login. Also, "role to group mapping" supports groups from a central LDAP-service.
- **Kerberos SSO with users only (method 4)**
  Users are logged in automatically with a ticket that they receive and cache during Windows- or Linux desktop login.
- **LDAP-based or Kerberos-based authentication (methods 1 to 4), with fallback to the local user database**
  If you configure R&S CHM for exclusively using LDAP and single sign-on authentication methods, the local users are no longer available on the web GUI. Only LDAP users or KDC users can access the web GUI for system status monitoring. However, you can configure a fallback method to the local user database if you specify the `builtin` key as an additional authentication method.

For detailed configuration examples, see the following `authentication` and `authorization` syntax descriptions.

To configure LDAP and single sign-on authentication

1. Under the host with the `[chm]` tag, configure the `authentication` key.
2. Configure user **authentication** as described under `authentication` on page 59.
3. Configure user **authorization** as described under `authorization` on page 63.

   You have configured R&S CHM for LDAP or SSO support. Web GUI users can log in to the web GUI with the users or user groups that are already configured on your network.

authentication (Authentication)

Configures the authentication method for all web GUI users.

Parameters:

monitoring — All authentication keys are specified under this key.
See monitoring on page 60.

---

monitoring (Authentication methods)

All authentication keys are specified under this key.

Parameters:

builtin — Built-in authentication method (fallback), see builtin on page 60.

gssapi — GSSAPI-based authentication methods, see gssapi on page 60.

ldap — LDAP-based authentication methods, see ldap on page 61.

---

builtin (Builtin authentication method)

Enables the local, built-in user database. If you specify this key, you can log in with the users "admin" and "operator" from the local user database when authentication using LDAP or SSO is not possible.

Default credentials

- **Operator**: *operator*, password *chmoperator*
- **Administrator**: *admin*, password *chmadmin*

If you only specify `builtin` without other authentication details, the configuration is equivalent to leaving out the `authentication` configuration at all, i.e. only the local users are available.

Example: **Local user database (method 0)**

Usage of the local user database (`builtin` specified):

```
authentication:
  monitoring:
    - builtin:
```

Usage of the local user database (`builtin` not specified):

```
authentication:
  monitoring:
```

Both authentication methods are equivalent.

---

gssapi (GSSAPI-based authentication method)

Configures the exchange of tokens as used for authentication methods "Kerberos SSO with SSSD for group information (method 3)" and "Kerberos SSO with users only (method 4)".

Parameters:

keytab <file_path>

Specifies the path to the key table (keytab) file.

Example: **SSO authentication variants**

SSO authentication only:

```
authentication:
  monitoring:
    - gssapi:
        keytab: /etc/opt/rohde-schwarz/chm/HTTP.keytab
```

SSO authentication with fallback to the local user database:

```
authentication:
  monitoring:
    - builtin:
    - gssapi:
        keytab: /etc/opt/rohde-schwarz/chm/HTTP.keytab
```

Example: **Kerberos SSO with SSSD for group information (method 3)**

This method retrieves user information from Kerberos tickets.
The LDAP group information is requested using the POSIX command id <user>. This command version uses the name service switch (NSS) to query group information through the privileged system security services daemon (SSSD) from the LDAP.
Only the SSSD reads the secret key table (keytab file) that contains a key for service principal. Only the SSSD connects to LDAP.
Specify the path to a valid *.keytab file. In this example, also the fallback login method builtin is configured:

```
authentication:
  monitoring:
    - builtin:
    - gssapi:
        keytab: /etc/opt/rohde-schwarz/chm/httpd.keytab
```

Example keytab file contents:

```
ktutil:  read_kt HTTP.chmserver.keytab
ktutil:  list
slot KVNO Principal
---- ---- ---------------------------------------------------
   1    2 HTTP/chmserver.your.org@YOUR.ORG
```

ldap (LDAP-based authentication method)

Obtains the credentials from a centrally maintained LDAP server.

Parameters:

server <FQDN> | <IP_address>

Specifies the address of the LDAP server, either its fully qualified domain name or its IP address. You can specify two redundant LDAP servers to enhance availability of this authentication method.

encryption ldaps | starttls

Configures the encryption method that is used to secure the communication between the LDAP server and the R&S CHM host.
The LDAP server must support your choice.

ldaps
Configures the *LDAP over SSL* protocol.

starttls
Configures the *LDAP over TLS* protocol.

base_dn

string

Specifies the LDAP distinguished name (DN) of the branch of the directory where the searches for users start from. The DN uniquely identifies an object in the Active Directory.

user_class

string

Specifies the LDAP class of user objects.

user_name_attr

string

Specifies the LDAP attribute that holds the user's name that is used for the login.

bind_dn

string

Specifies the DN used to bind to the server when searching for users. Only necessary for authentication method "LDAP with dedicated bind user (method 1)", see following example.

bind_pwd_path

string

Path of the LDAP password within the R&S CHM password store. Only necessary for authentication method "LDAP with dedicated bind user (method 1)".
See also: Section 6.6, "Managing password identifiers", on page 66

Example:

LDAP with dedicated bind user (method 1)
This method requires a user name and the path of the authentication password for the bind operation. A dedicated bind user authenticates itself against LDAP.
Specify `bind_dn` and `bind_pwd_path`:

```
authentication:
  monitoring:
    - ldap:
        server: [ldapserv.ourlocal.net, ldapserv2.ourlocal.net]
        encryption: ldaps
        base_dn: ou=Foo_Users,dc=foo,dc=bar,dc=baz
        user_class: user
        user_name_attr: sAMAccountName
        bind_dn: user
        bind_pwd_path: ldap/icinga_ldap_user
```

Example: **LDAP anonymous bind (method 2)**

This method does not require user credentials at all and accesses the LDAP as anonymous. To use this method, it is necessary that you explicitly allow this binding method on the LDAP server.

Omit both keys `bind_dn` and `bind_pwd_path` or leave them empty:

```
authentication:
  monitoring:
    - ldap:
        server: [ldapserv.example.net, ldapserv2.example.net]
        encryption: ldaps
        base_dn: ou=ldap_users,dc=ldapserv,dc=ourlocal,dc=net
        user_class: user
        user_name_attr: sAMAccountName
```

---

authorization (Authorization)

Configures the authorization method for all web GUI users.

Parameters:
monitoring

roles — string

Specifies and configure the user roles that are available. You can choose the names freely, e.g. `administrators` and `operators`. The specified roles are generated on the R&S CHM host.

permissions — acknowledge | check | comment | downtime | maps | statusview

List of permissions that are assigned to the role (optional).

acknowledge
Acknowledge hosts or service problems by selecting the ✎"Acknowledge" button on the web GUI.

check
Start a check immediately by selecting the ↻"Check now" button on the web GUI.

comment
Leave a comment for a host or service by selecting the 🗨 "Comment" button on the web GUI.

downtime
Schedule a downtime by selecting the ⏻ "Downtime" button on the web GUI. Host or service problems do not show up for the dedicated host or service during the downtime.

maps
View maps on the web GUI. See Section 6.8, "Configuring graphical system views (maps)", on page 72.

statusview
Adds a "Status View" page to the web GUI > "Overview" menu. Thus, you can provide a page on the web GUI that sorts all the hosts via the host group. Configure the hosts groups for your hosts accordingly, see `hosts` on page 41 > `hostgroups`.

checkdetails
Provides more detailed information on the web GUI about check execution, such as command, check source, last update, next update and check attempts. This information can be useful for administrators.

manual
In combination with a passive check configuration, this permission creates a button for "passive" checks on the web GUI. See also: `manual` on page 122

users

list of users

List of users to which R&S CHM applies the role, e.g. `admin`, `operator`, `john` (optional).

groups

list of groups

List of user groups to which R&S CHM applies the role, e.g. `company_chm_admins` (optional).

Example:

LDAP with dedicated bind user (method 1)
Example with LDAP `users`:

```
authorization:
  monitoring:
    roles:
      operators:
         permissions:
           - acknowledge
           - comment
           - check
         users:
           - chm_operator
           - chm_monitor
```

Example with LDAP `groups`:

```
authorization:
  monitoring:
    roles:
      administrators:
         permissions:
           - acknowledge
           - comment
           - downtime
         groups:
           - company_chm_admins
```

Example: **Kerberos SSO with users only (method 4)**

This method retrieves user information from Kerberos tickets. You require a configured key distribution center (KDC) in your system.

Group information is not included in Kerberos tickets. As a consequence, you cannot use groups for authorization if only Kerberos without LDAP is available due to security restrictions.

Specify `permissions` for `users`:

```
authorization:
  monitoring:
    roles:
      operators:
        permissions:
          - acknowledge
          - comment
          - check
        users:
          - chm_operator@domain.org
          - chm_monitor@domain.org
```

Example: **Combined authentication method: Kerberos SSO with SSSD for group information (method 3) with fallback to local user database (method 0)**

This example also configures the `builtin` fallback authentication method. The user `admin` is the fallback user.

```
authentication:
   monitoring:
     - builtin:
     - gssapi:
         keytab: /etc/opt/rohde-schwarz/chm/HTTP.keytab
authorization:
   monitoring:
     roles:
       commenter:
         permissions:
           - comment
         users:
           - johndoe@RSINT.NET
           - admin
       downtimer:
         permissions:
           - downtime
         groups:
           - domainoperators
```

Example: **Local user database (method 0)**

`builtin` authentication method in combination with authorization:

```
authentication:
  monitoring:
    - builtin:
authorization:
  monitoring:
    roles:
      commenter:
        permissions:
          - comment
        users:
          - operator
```

6.6 Managing password identifiers

All passwords for communication between R&S CHM and an LDAP server or R&S CHM and the monitored services are encrypted using GPG. To ease password handling, R&S CHM provides a **password manager**.

The password manager lets you safely specify necessary password identifiers for communication of R&S CHM via the following interfaces:

- LDAP simple authentication password
- SNMP
- Proprietary interfaces, e.g. VMware

Change all passwords in the password store before your system goes live.

To list all password identifiers

Access authorization: *root*

1. Log in to the R&S CHM host.
2. Enter the following command:
   ```
   # chmpass ls
   ```
   The currently defined password identifiers are listed. For an example output, see the following example.

Example: List configured password identifiers

```
$ chmpass ls
Password Store
├── tiger
├── bumblebee
└── ant
```

To add a password identifier

Access authorization: *root*

1. Log in to the R&S CHM host.
2. Type the following command:
   `# chmpass insert <password_identifier>.`
   Example: `# chmpass insert tiger`
3. Enter the password identifier.
4. Repeat the password identifier.

   You successfully added the password identifier.

To remove a password identifier

Access authorization: *root*

1. Log in to the R&S CHM host.
2. Enter the following command:
   `# chmpass rm <password_identifier>`
3. Confirm deletion.

   You successfully removed the specified password identifier.

Example: Delete a password identifier

The name of the identifier is "tiger".

```
$ chmpass rm tiger
Are you sure you would like to delete tiger? [y/N] y
removed ‘/var/opt/chm/password-store//tiger.gpg’
```

To set a password identifier in the configuration file

Access authorization: *root*

1. Access the `chm.yaml` file.
   See also: "To access the configuration file" on page 40
2. Under the `check:` key for the resource, add the key-value pair:
   `<identifier>: <password_identifier>`
   Examples
   - For snmpv3: `snmp_connection` > `secname: tiger`
   - For vmware: `user: lion`

   R&S CHM can access the checked resources via the set password identifier.

6.7 Configuring R&S RAMON for monitoring

This monitoring method uses a gRPC-based R&S CHM service called `chmrd`. It replaces the deprecated Windows SNMP service.

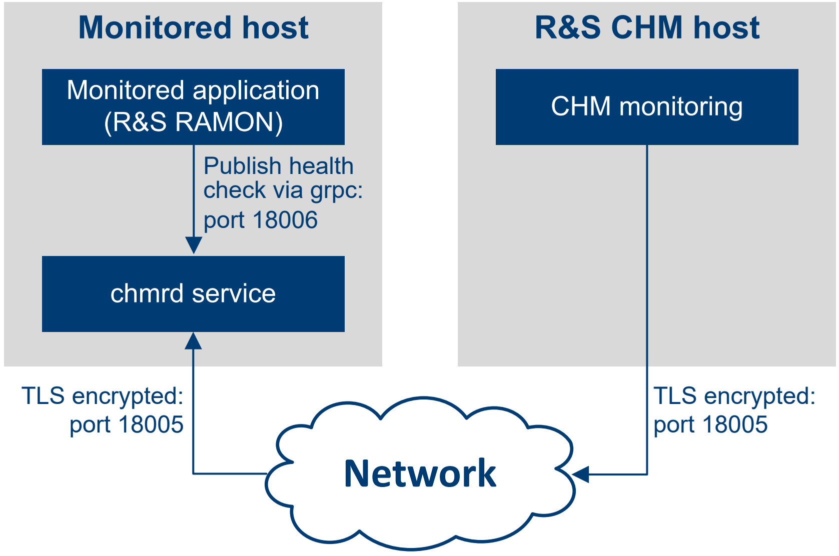

Monitored host
Monitored application (R&S RAMON)
Publish health check via grpc: port 18006
chmrd service
TLS encrypted: port 18005
R&S CHM host
CHM monitoring
TLS encrypted: port 18005
Network

Figure 6-9: Monitoring of applications, e.g. R&S RAMON

The following description explains the monitoring steps visualized in the previous figure.

Monitored application and R&S CHM

Applications "publish" monitoring data to `chmrd`. R&S CHM fetches the monitoring data from `chmrd`.

The chmrd service

The service `chmrd` gathers monitoring data sent by applications and makes it available to R&S CHM instances. Currently, it has to be installed on the same Windows host that also runs the monitored application. It is necessary that you install `chmrd_<version>.msi` on the agent that runs R&S RAMON, see Section 4.2.1, "Installing Windows agents", on page 23.

Interface definition

The service provides a gRPC interface that can be used to both send and query monitoring data. The interface is defined in a protobuf file. This file describes the services provided by `chmrd` and the data model that is used for communication and even how this data is serialized on the wire. The file thus takes the role of a serialization document.

Security aspects

The chmrd service uses two separate TCP ports:

- For communication with clients on the same host: local port, default port number 18006
  On the local port, the service only listens for connections from localhost. There is no encryption or authentication or authorization when using the local port. Its main use case is for communication between chmrd and the monitored application.
- For clients on remote hosts: remote port, default port number 18005.
  When communicating over the remote port, chmrd enforces TLS encryption and client authentication using X.509 certificates to secure network communication. There is no authorization mechanism in place yet which means an authenticated client is allowed to both send and query monitoring data without any restrictions.

6.7.1 Configuring the chmrd service

The only officially supported way of configuring chmrd is to pass command-line arguments to the service.

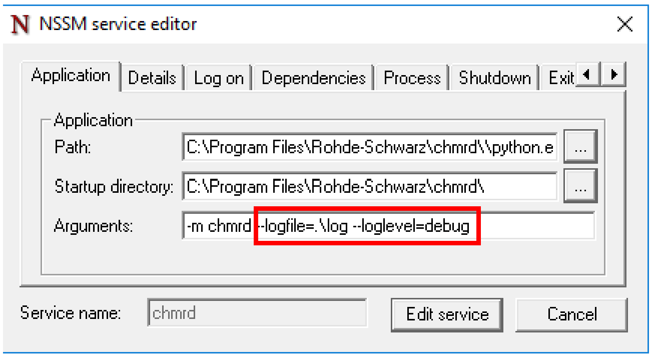
Typically, you can use the default chmrd configuration. However, if you need to change the configuration, continue as described in the following procedure.

To configure the chmrd service

This procedure assumes that the chmrd software is already installed.

1. Open the installation directory:
   `C:\Program Files\Rohde-Schwarz\chmrd\`
2. Open a command prompt window in the installation directory.
3. Run the following command:
   `.\nssm.exe edit chmrd`

   The "NSSM service" editor opens.



Path: C:\Program Files\Rohde-Schwarz\chmrd\\python.e

Startup directory: C:\Program Files\Rohde-Schwarz\chmrd\

Arguments: -m chmrd --logfile=.\log --loglevel=debug

Service name: chmrd

4. Configure the desired arguments as listed in Table 6-4.

   Note: Always keep the "-m chmrd" argument. This information tells the python interpreter which module to use to start the service.

Table 6-4: Command-line arguments for configuring the chmrd service

Argument (short)	Argument (long)	Default value	Description
"-a"	"--address"	"0.0.0.0"	IP address the server runs on
"-p"	"--port"	"18005"	Port for connections from remote hosts
"-P"	"--local-port"	"18006"	Port that clients on localhost can use without needing to authenticate themselves
"-d"	"--cert-dir"		Directory with certificates and keys
"-C"	"--server-cert"		Server certificate path
"-R"	"--server-root-cert"		Server root certificate path
"-K"	"--server-priv-key"		Server-private key path
"-c"	"--client-root-cert"		Client root certificate path
	"--insecure"		If set, disable encrypted message transport and server/client authentication (without a value)
	"--loglevel"	"info"	One of "debug", "info", "warning", "error", "critical"
	"--logfile"		Logfile path
	" -- logfilemode"	"w"	"a" or "w" "a" for appending to log file. "w" for truncating log file and starting a new one when the service is restarted

Example:

The following arguments set specific ports and how the log file is treated.

"-m chmrd -p=18007 -P=18008 --logfilemode=a"

About certificates and keys

All certificates and keys used for `chmrd` have to be PEM encoded.

To achieve encrypted and authenticated network communication, `chmrd` needs the following:

- **A server certificate chain**
  A certificate chain is a list of certificates where the issuer of each one of them matches the subject of the following. Also each certificate - except for the last - is signed with the secret key corresponding to the next certificate. The last certificate in the chain is self-signed, which makes it a root certificate.
  Usually, this chain consists of a certificate issued for the host on which the service is running. This certificate is followed by some root CA's certificate that was used to issue the certificate of the host. You can specify these two parts of the certificate chain by using the "--server-cert" and the "--server-root-cert" arguments, including the path to the corresponding files.
  For the uncommon use case that the chain consists of more than two certificates, you can split up the certificates to the two files specified by "--server-cert" and "--

server-root-cert". Make sure that the resulting chain fulfills the criteria for a certificate chain described above.

- **A server-private key**
  This key is the private key corresponding to the certificate on the server, i.e. the monitored host used for encrypting network communication. The file containing the key can be specified by using the "--server-priv-key" argument.
- **A client root certificate**
  `chmrd` expects remote clients to provide a certificate chain to authenticate themselves. You can specify a file containing one or more root certificates for these chains by using the "--client-root-cert" argument.

Default paths for certificates and keys

There are different possible combinations of how to use command-line arguments in `chmrd`. The following tables list the defaults that are used in the different cases **A**, **B** and **C**.

A) If no command-line arguments are specified, the defaults use the fully qualified domain name (FQDN) of the host, see the following table.

Table 6-5: No command-line arguments are specified

Argument (long)	Default value
"--server-cert"	`C:\ProgramData\icinga2\var\lib\icinga2\certs\<FQDN>.crt`
"--server-priv-key"	`C:\ProgramData\icinga2\var\lib\icinga2\certs\<FQDN>.key`
"--server-root-cert"	`C:\ProgramData\icinga2\var\lib\icinga2\certs\ca.crt`
"--client-root-cert"	`C:\ProgramData\icinga2\var\lib\icinga2\certs\ca.crt`

The default file locations here correspond to the certificate settings you usually already made for the R&S CHM Windows agent, see Section 5, "Deploying certificates", on page 32. Thus, no extra configuration is necessary for the `chmrd` service. Also, the `chmrd` service expects that both server and clients to use same root certificate by default.

B) If you specify "--cert-dir", you can set a custom location for all certificates and key, see the following table.

Table 6-6: Only --cert-dir is specified

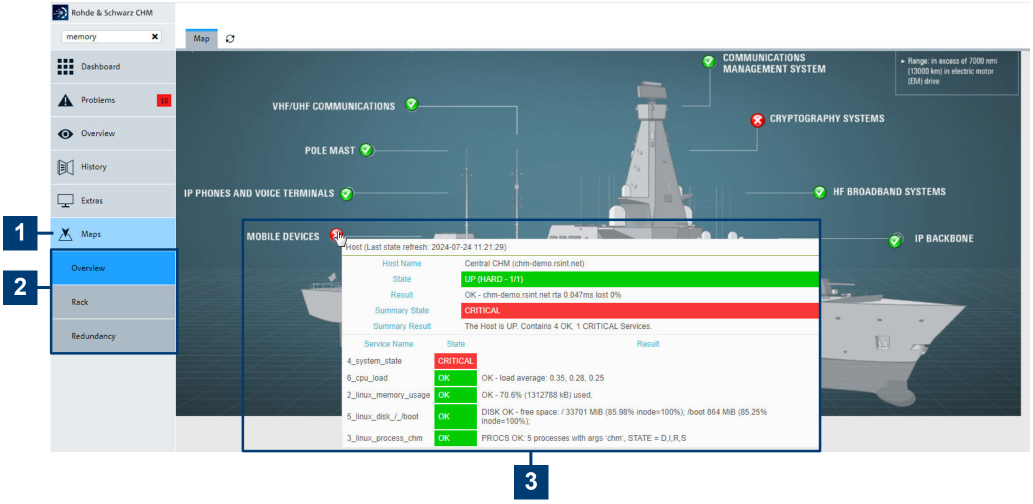
Argument (long)	Default value
"--server-cert"	`<CERT_DIR>\<FQDN>.crt`
"--server-priv-key"	`<CERT_DIR>\<FQDN>.key`
"--server-root-cert"	`<CERT_DIR>\ca.crt`
"--client-root-cert"	`<CERT_DIR>\ca.crt`

C) If one or all "--server-cert", "--server-root-cert", "--server-priv-key", "--client-root-cert" are specified, you can always specify a customized, absolute path to certificates and key.

For information about configuration of the check in the `chm.yaml` file, see `chm_remote_grpc` on page 101.

6.8 Configuring graphical system views (maps)

You can add and configure graphical elements to R&S CHM. These elements let you visually track the system's status on customizable maps, providing a more intuitive and comprehensive understanding of the system's operation. After the configuration in the `chm.yaml` file, you can find all configured graphical system views on the web GUI under "Maps" (**1**, **2**). R&S CHM lets you visualize the status of hosts, services, host groups or service groups.



Host (Last state refresh: 2024-07-24 11:21:29)

Host Name	Central CHM (chm-demo.rsint.net)
State	UP (HARD - 1/1)
Result	OK - chm-demo.rsint.net rta 0.047ms lost 0%
Summary State	CRITICAL
Summary Result	The Host is UP. Contains 4 OK, 1 CRITICAL Services.

Service Name	State	Result
4_system_state	CRITICAL	
6_cpu_load	OK	OK - load average: 0.35, 0.28, 0.25
2_linux_memory_usage	OK	OK - 70.6% (1312788 kB) used.
5_linux_disk_/_/boot	OK	DISK OK - free space: / 33701 MiB (85.98% inode=100%); /boot 864 MiB (85.25% inode=100%);
3_linux_process_chm	OK	PROCS OK: 5 processes with args 'chm', STATE = D,I,R,S

Figure 6-10: Graphic system views (maps)

1 = "Maps" main menu
2 = Individual "Maps" views
3 = Mouse over on the status icon provides details. Select the status icon to navigate to the configured host or service.

All example figures here explain the general behavior but do not reflect true systems or subsystems.

To prepare the background images

Each map is composed of a background image, a status icon and an optional label. To determine the coordinates for the status icons on an image, you can use almost any image editor. The labels are automatically filled with the `displayname` or `name` of that host or service. The label is shown to the right of a status icon.

1. Save the background images as pixel graphics of type PNG or JPEG in the correct, final size and resolution. We recommend using images in 96x96 pixels resolution.

   Note: R&S CHM does not modify or adapt the image size.

2. Upload the images to the R&S CHM host using WinSCP or LCSM.

`/etc/opt/rohde-schwarz/chm/maps/`

3. Determine the (x,y)-coordinates for the **status icons** that you want to show on the images. The following example shows how you use Microsoft Paint to determine the coordinates for the status icons on the graphic.

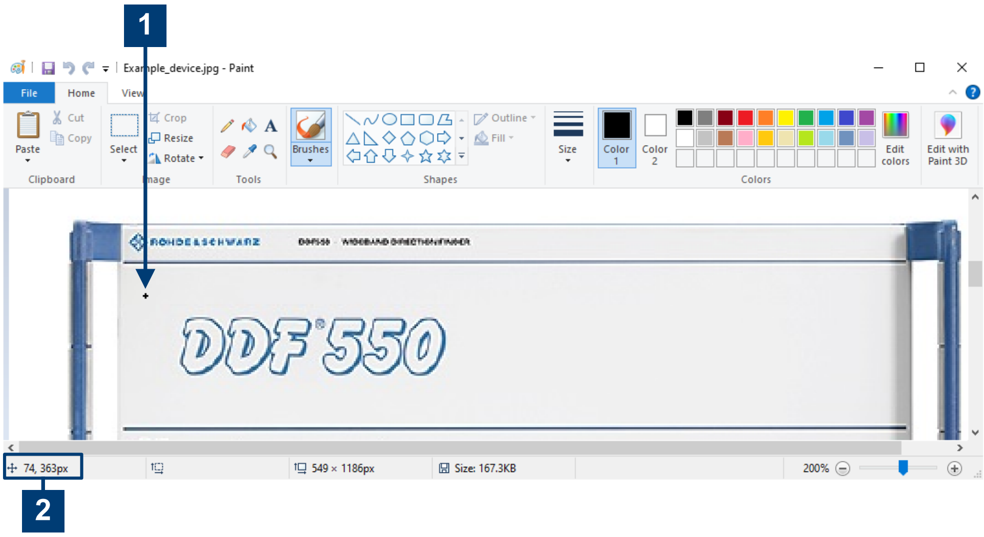


Figure 6-11: Coordinates of a cursor position

1 = Pointer location
2 = Status bar with the coordinates, e.g. "74, 363px"

a) Open the graphic in the editor.
b) If necessary, turn on the status bar.
c) Point to the location where you want to insert status icon (**1**). The location marks the top-left position of the status icon.
The first value is the x-value and the second is the y-value (**2**).

To configure graphical system views (maps)

1. Open the `chm.yaml` file. See Section 6.3, "Changing the configuration", on page 40.
2. Enter the coordinate value pairs (see step 3) in the `maps` > `x` and `y` keys of the corresponding hosts and services. Optionally, you can configure item-specific label formats.
For syntax details, see `maps` on page 93.
3. Configure the top-level `maps` key in the `chm.yaml` file. Here, you specify the background image and label layout.

   You can configure the label layout for each map separately:
   - The name that appears on the R&S CHM GUI
   - The image filename
   - The label format: background color, label border and label style. Also, you can hide all labels on a map.

For syntax details, see `Graphical system view (maps)` on page 74.

When finished, you can view the result on the web GUI.

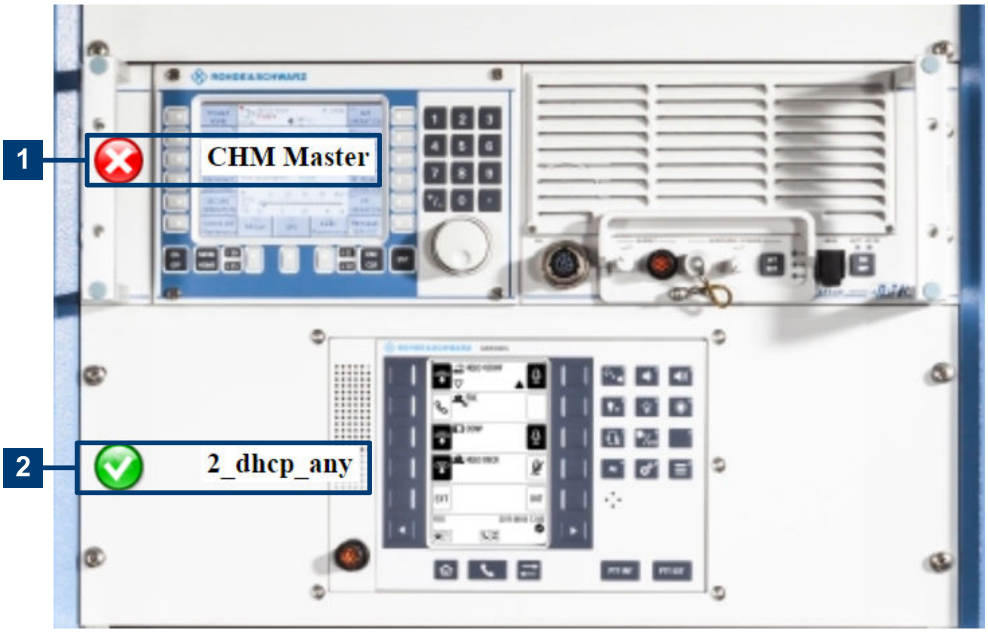


Figure 6-12: Map example with status icons and labels

1 = "CHM Master" host, status "CRITICAL"
2 = "2_dhcp_any" service, status "OK"

Graphical system view (maps) maps

Configures all graphical elements for visualization of the system status on the R&S CHM GUI. To view the maps on the web GUI, users need the `maps` permission.

Related parameters

- `maps` on page 93
- Section 6.5, "Configuring web GUI users", on page 57

Parameters:

name
: string
Name of the individual subsystem on the GUI. For an example, see Figure 2-2.

background_image
: file name
Filename of the background image, e.g. `my_system.jpg`. R&S CHM supports the file types PNG or JPEG.

label_show
: True | False
Visibility of labels on the GUI (optional). If no key is specified, the labels are shown (`True`).

False

Hides the labels on the GUI

label_background

hex string

Map-specific background color of the label (optional).
Specifies the colors for graphical elements on the GUI in Hex code values. Table 6-7 lists basic colors that you can start with (from the West Library/Texas Weslyan University, page retrieved 2024-02-06). On the internet, you can find multiple pages that let you pick the color codes, e.g. HTML color codes.

label_border

hex string

Map-specific border color of the label (optional).

label_style

string

Map-specific font family, font weight and font size of the label text (optional).
Standard CSS font families: `serif`, `sans-serif`, `cursive`, `system-ui`.
Standard font weights: `normal`, `bold`, `lighter`, `bolder`, `<font-weight-absolute>` (numeric values between 1 and 1000)

Example:

Three individual `maps` configurations. Specify the `maps` key on the same indention level as the `hosts` key.

```
maps:
  - name: Overview
    background_image: ship1.jpg
    label_show: False
  - name: Rack
    background_image: rack1.jpg
    label_background: "#FFFFFF"
    label_border: "#FFFFFF"
    label_style: "font-family:sans-serif;\
color:#000000;font-weight:bold;font-size:20;"
  - name: Redundancy
    background_image: redundancy1.jpg
    label_background: "#FFFFFF"
    label_border: "#FFFFFF"
    label_style: "font-family:sans-serif;color:#000000;\
font-weight:bold;font-size:20;"
```

Table 6-7: Basic color codes

Color	Hex code
Black	#000000
White	#FFFFFF
Red	#FF0000
Lime	#00FF00
Blue	#0000FF
Yellow	#FFFF00

Color	Hex code
Cyan/Aqua	#00FFFF
Magenta/Fuchsia	#FF00FF
Silver	#C0C0C0
Gray	#808080
Maroon	#800000
Olive	#808000
Green	#008000
Purple	#800080
Teal	#008080
Navy	#000080

6.9 Configuring the SNMP upstream interface

R&S CHM provides an SNMP-based interface that lets you query an aggregated system state from upstream monitoring solutions. The SNMP upstream interface provides information about the names, states and last-change timestamps for the overall system and its host groups. R&S CHM listens for incoming SNMP requests on port 161/udp on all available network interfaces. Currently, R&S CHM only supports SNMP version 2c. The system name is "RS-CHM" (fixed).

6.9.1 Activating the interface

You can activate the SNMP upstream interface by adding the `snmp_connection` key to the master R&S CHM host instance in the `chm.yaml` file. The only available setting is the value for the SNMP `community` string, which must be passed to R&S CHM to retrieve data from the interface.

For details about the state aggregation logic, see Section 6.2, "Understanding aggregated states", on page 39.

For a detailed description of the management information, see the management information base (MIB) files. All MIBs are contained in the delivery.

Example: Configuration of the SNMP upstream interface

Here, the value testcommunity must be used to retrieve data from the interface. The checks are omitted.

```
hosts:
  - name: host1.de
    tags: [chm]
    hostgroups: [saturn]
    snmp_connection:
      community: testcommunity
    checks:
      ...
```

In your master R&S CHM host configuration, substitute this example community string by a custom community string.

See also: snmp_connection on page 95

6.9.2 Configuring SNMPv2 traps

You can configure R&S CHM to inform a list of SNMP notification receivers about system status changes via SNMPv2 traps. To do so, add the snmp_connection > trapreceivers key to the master R&S CHM host instance in the chm.yaml file. The traps contain the system state and all host group states.

For a detailed description of the management information, see the management information base (MIB) files. All MIBs are contained in the delivery.

Example: Configuration of SNMPv2 traps

For each SNMP notification receiver, specify the host and port to which R&S CHM sends the trap. Also, specify the SNMP community that is expected by the SNMP notification receiver to accept the trap.

```
hosts:
  - name: host1.de
    displayname: CHM master
    tags: [chm]
    snmp_connection:
      community: foo
      trapreceivers:
        - host: host2.de
          port: 162
          community: bar
        - host: host3.de
          port: 1162
          community: baz
```

See also: snmp_connection on page 95

6.10 Configuring distributed monitoring

This chapter helps you configure different variants of system status monitoring. Distributed monitoring means that you configure multiple R&S CHM instances that either monitor other hosts or devices, or that send monitoring results to R&S CHM hosts.

Such configurations can comprise a second, redundant R&S CHM host or multiple R&S CHM hosts that are distributed over different subsystems. A subsystem is at least one R&S CHM node that is grouped with any number of non-R&S CHM hosts or devices, or both. Each R&S CHM host instance in a subsystem provides its own web GUI.

In the following description, involved R&S CHM instances are named by their role in the status monitoring system.

- A **master** is an R&S CHM host instance that is located in the top-level subsystem. A master receives the results of all checks that are executed by itself and the check results from subordinated satellites and agents.
- A **satellite** is an R&S CHM host instance that is not placed in the top-level subsystem. A satellite sends its check results to configured master instances. The web GUI of the satellite only shows the check results of the satellite and subordinated agents. Check results from other satellite instances or master instances are unavailable.
- An **agent** is an R&S CHM agent instance on Linux or Windows hosts. An agent only checks itself and sends the results to a parent satellite or master. An agent does not provide an own web GUI.

Simple monitoring configuration

Typically, you configure a monitoring setup that comprises a single master R&S CHM host and multiple Linux and Windows agents.

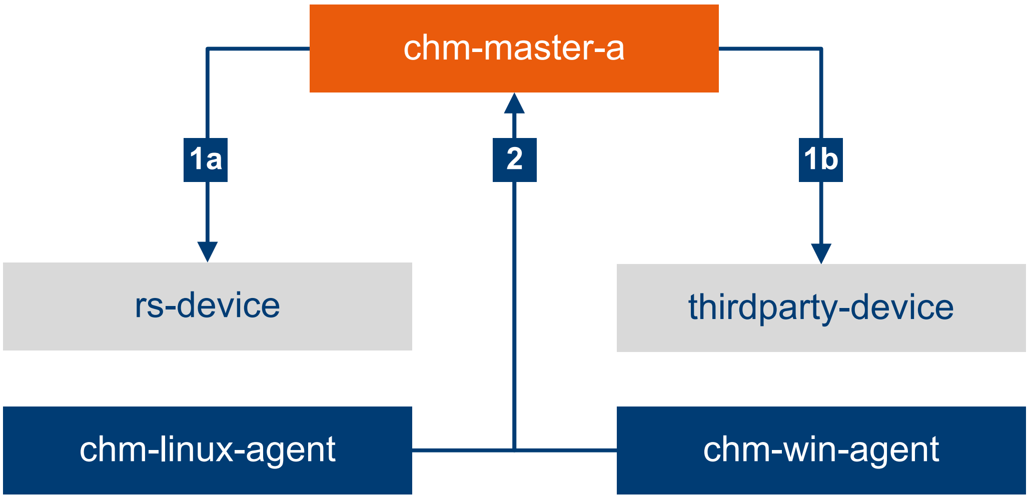


Figure 6-13: Simple monitoring configuration

1a, 1b = Master monitors devices.
2 = Agents send monitoring results to master.

The previous figure shows a monitoring configuration where a master, i.e. the R&S CHM host, monitors all kinds of devices and hosts that are not acting as an agent (**1a, 1b**). The agents monitor the hosts on which they are installed. The agents send their monitoring results to the R&S CHM host (**2**). In this configuration, the web GUI of the R&S CHM host shows all monitored hosts and services.

The following chapters explain how you configure monitoring variants that use multiple R&S CHM host instances in parallel.

6.10.1 Configuring high availability monitoring

For high availability (HA) monitoring, you configure a second R&S CHM host as a **secondary master**. Such a configuration provides the following features:

- **Data synchronization**: Both masters synchronize all monitoring data between each other, and they let you view to whole system state independently.
- **Data duplication**: Both masters save all monitoring data to their own local database. Due to this mechanism, you can profit from an automatically created backup.
- **Failover**: If one master becomes unavailable, the R&S CHM agents automatically send their monitoring data to the remaining, intact master.
  The automatic failover procedure avoids a single point of failure for receiving the check results from R&S CHM agents at master level.

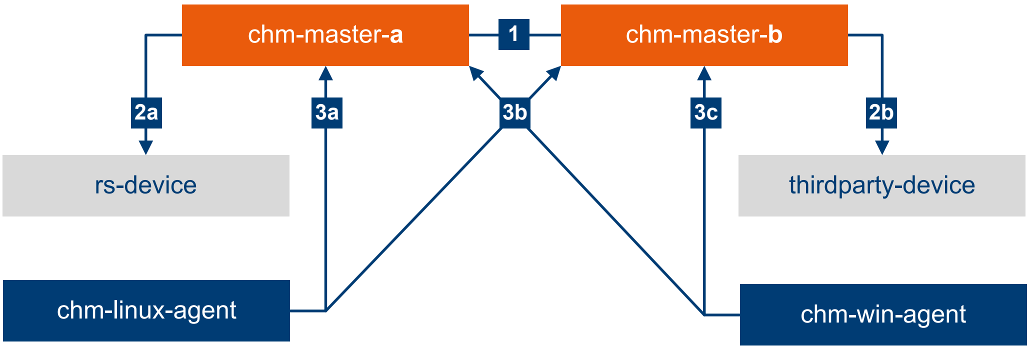


Figure 6-14: High availability monitoring

1 = Synchronization of monitoring results between masters.
2a, 2b = Masters monitor devices.
3a, 3b, 3c = Agents send monitoring results to masters.

To set up a HA monitoring system

1. On both masters, install the R&S CHM host software.
   How to: Section 4, "Installing R&S CHM", on page 21
2. Install certificates and keys.

How to: Section 6.10.5, "Deploying certificates for distributed monitoring", on page 92

3. Edit the `chm.yaml` file to describe the HA monitoring configuration.
   How to: Section 6.10.1.1, "Editing the YAML configuration for HA monitoring", on page 80.
4. Both masters require an identical `chm.yaml` file. Save this file here:
   `/etc/opt/rohde-schwarz/chm/`
5. Restart the chm service on both masters to take the changes effect:
   `# systemctl restart chm`.

6.10.1.1 Editing the YAML configuration for HA monitoring

HA monitoring configurations require two R&S CHM host instances, one instance serves as the primary master the other instance serves as the secondary master.

1. Specify the entries in the `chm.yaml` file for the HA monitoring configuration.
   a) Under the `hosts` configuration of the primary master, add the host configuration of the secondary master.
   b) Configure the secondary host as the high availability master:
      `tags: ["icinga2_ha"]`
2. Except for masters or agents, you can configure a relationship to the secondary master. To do so, add this key:
   `checked_by: "<HA-master-fqdn>"`

Example: YAML configuration: HA monitoring

This example:

- Uses the host names from Figure 6-14
- Omits any checks for clarity

```
hosts:
  # primary master
  - name: "chm-master-a"
    tags: ["chm"]

  # secondary master
  - name: "chm-master-b"
    tags: ["icinga2_ha"]

  # linux agent
  - name: "chm-linux-agent"
    connections: ["icinga2_linux"]

  # windows agent
  - name: "chm-win-agent"
    connections: ["icinga2_win"]

  # devices
  - name: "rs-device"
  - name: "thirdparty-device"
    checked_by: "chm-master-b"
```

6.10.1.2 Configuring R&S CHM agents for HA monitoring

It is necessary that you inform the agents about the existence of both masters.

► Run these scripts to complete agent configuration:

- On Linux agents, run the `chm_node_setup` shell script.
- On Windows agents, run the `chm-node-setup.bat` batch script.

For parameterization see the following examples that use the FQDNs from Figure 6-14.

Example:

Script on the Linux agent **chm-linux-agent**:

```
chm_node_setup \
--subsys chm-linux-agent \
--parent-subsys chm-master-a \
--parent-chm chm-master-a \
--second-parent-chm chm-master-b
```

Example:

Script on the Windows agent **chm-win-agent**:

```
"C:\Program Files\chm\chm-node-setup.bat" \
--subsys chm-win-agent \
--parent-subsys chm-master-a \
--parent-chm chm-master-a \
--second-parent-chm chm-master-b
```

6.10.2 Configuring subsystems

You can subdivide a status monitoring system into multiple subsystems for multi-level monitoring purposes. Subsystems then define the structure of the overall system. Also, subsystems define the relations between hosts, i.e. the hosts that are directly monitored by an R&S CHM host and the R&S CHM hosts that synchronize check results.

How to configure subsystems:

- Section 6.10.3, "Configuring multi-level monitoring", on page 83
- Section 6.10.4, "Configuring multi-level HA monitoring", on page 88

In the `chm.yaml` file, you add subsystems on the same indentation level as hosts.

subsystems (Subsystems for multi-level monitoring)

Defines the subsystems of the status monitoring system.

Parameters:

name	string Specifies the name of the subsystem.
hosts	list of strings Specifies the members of a subsystem using their host names.
parent_subsystem	For subordinated subsystems only, specify the related higher-level subsystem.

Example: This example shows a small excerpt for orientation purposes.

```
subsystems:
  - name: "A"
    hosts:
      - "chm-master-a"

  - name: "B"
    hosts:
      - "chm-satellite-b"
      - "rs-device"
    parent_subsystem: "A"

  - name: "C"
    hosts:
      - "chm-satellite-c"
      - "thirdparty-device"
    parent_subsystem: "A"
```

For comprehensive examples, see the following chapters.

6.10.3 Configuring multi-level monitoring

For multi-level monitoring, you configure more than one R&S CHM instance in a tree-like structure with three or more monitoring levels. A multi-level configuration provides these features:

- **Subsystem monitoring**: Split up the system into subsystems. Each R&S CHM node only monitors its subtree of the system.
- **Monitoring of remote systems**: For distant system components, a tree-like configuration reduces network traffic between remote locations and also helps reduce the load on the top-level master.

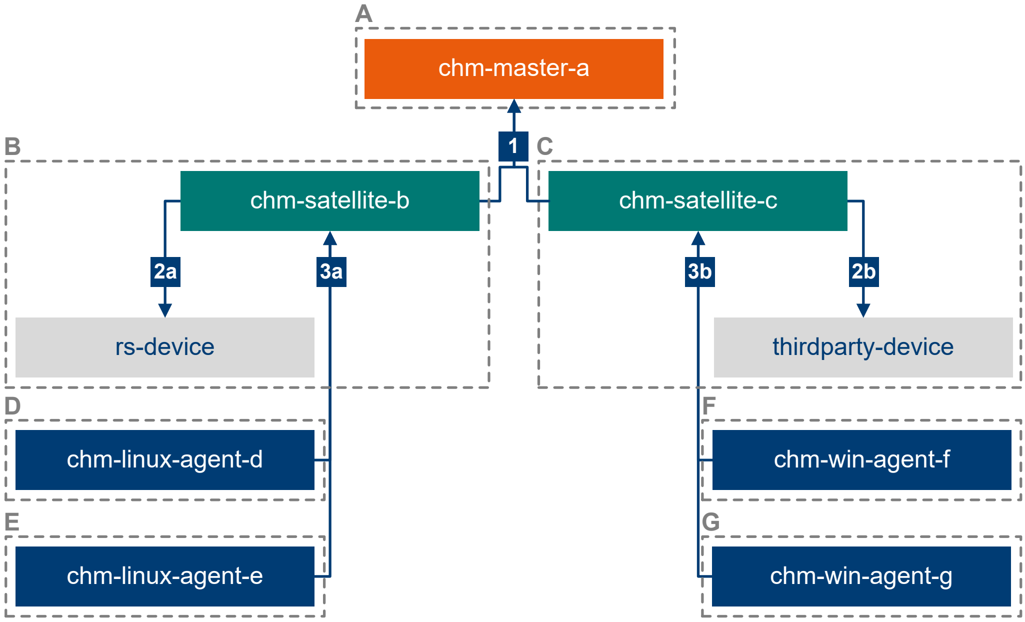


Figure 6-15: Multi-level (three-level) monitoring example

1	=	Satellites send subsystem monitoring results to master.
2a, 2b	=	Satellites in subsystems monitor devices.
3a, 3b	=	Agents send monitoring results to satellites.

The status monitoring system in the previous figure is subdivided in subsystems **A** to **G**.

In sum, the system contains three R&S CHM host instances, i.e. one each in subsystems A, B and C. The R&S CHM hosts adopt the following roles:

- The **master** is the R&S CHM host instance in top-level subsystem **A**.
- The **satellites** are R&S CHM host instances in second-level subsystems **B** and **C**.

Each R&S CHM host instance provides its own web GUI. Thus, you can monitor the following on these web GUIs:

- chm-master-a (subsystem **A**): Monitor the whole system.
- chm-satellite-b (subsystem **B**): Monitor subsystems **B**, **D** and **E**.
- chm-satellite-c (subsystem **C**): Monitor subsystems **C**, **F** and **G**.

The remaining R&S CHM nodes are four R&S CHM agents, i.e. one each in subsystems **D**, **E**, **F** and **G**.

To set up a multi-level monitoring system

1. On the master and the satellites, install the R&S CHM host software.
   How to: Section 4, "Installing R&S CHM", on page 21
2. On each other host that masters or satellites cannot monitor with external checks, install the agent software.
   How to: Section 4.2, "Installing R&S CHM agents", on page 23
3. Install certificates and keys.

How to: Section 6.10.5, "Deploying certificates for distributed monitoring", on page 92

4. Edit the `chm.yaml` file to describe the multi-level monitoring architecture.
   How to: Section 6.10.3.1, "Editing the YAML configuration for multi-level monitoring", on page 85.
5. All masters and satellites require an identical `chm.yaml` file. Save this file here:
   `/etc/opt/rohde-schwarz/chm/`
6. Restart the chm service on all masters and satellites to take the changes effect:
   ```
   # systemctl restart chm
   ```
   We recommend starting the service in sequence on the master and then on the satellites.
7. On each agent, run the node setup scripts with options that describe the multi-level system. See Section 6.10.3.2, "Configuring agents for multi-level monitoring", on page 87.

6.10.3.1 Editing the YAML configuration for multi-level monitoring

Multi-level monitoring configurations require that you configure the `subsystems` key above the `hosts` key.

► Specify the entries in the `chm.yaml` file for the multi-level monitoring configuration:
- The names of all subsystems
- The members of the subsystems, i.e. masters, satellites or agents, or monitored hosts or devices
- Exactly one parent subsystem except for the top-level subsystem

Example: YAML configuration: multi-level monitoring

A satellite always requires a R&S CHM host installation on Linux. Thus, the satellites require the `connections: ["icinga2_linux"]` key.

This example:

- Uses the host names from Figure 6-15
- Omits any checks for clarity

```
subsystems:
  - name: "A"
    hosts:
      - "chm-master-a"

  - name: "B"
    hosts:
      - "chm-satellite-b"
      - "rs-device"
    parent_subsystem: "A"

  - name: "C"
    hosts:
      - "chm-satellite-c"
      - "thirdparty-device"
    parent_subsystem: "A"

  - name: "D"
    hosts:
      - "chm-linux-agent-d"
    parent_subsystem: "B"

  - name: "E"
    hosts:
      - "chm-linux-agent-e"
    parent_subsystem: "B"

  - name: "F"
    hosts:
      - "chm-win-agent-f"
    parent_subsystem: "C"

  - name: "G"
    hosts:
      - "chm-win-agent-g"
    parent_subsystem: "C"

hosts:
  # master in A
  - name: "chm-master-a"
    tags: ["chm"]
```

```
# satellite in B
- name: "chm-satellite-b"
  connections: ["icinga2_linux"]

# satellite in C
- name: "chm-satellite-c"
  connections: ["icinga2_linux"]

# linux agent in D
- name: "chm-linux-agent-d"
  connections: ["icinga2_linux"]

# linux agent in E
- name: "chm-linux-agent-e"
  connections: ["icinga2_linux"]

# linux agent in F
- name: "chm-win-agent-f"
  connections: ["icinga2_win"]

# linux agent in G
- name: "chm-win-agent-g"
  connections: ["icinga2_win"]

# devices
- name: "rs-device"
- name: "thirdparty-device"
```

6.10.3.2 Configuring agents for multi-level monitoring

It is necessary that you inform the agents about these relations:

- The own subsystem.
- The parent subsystem.
- The connection to master or satellite.

To configure the agents

► Run these scripts to complete agent configuration:

- On Linux agents, run the `chm_node_setup` shell script.
- On Windows agents, run the `chm-node-setup.bat` batch script.

For parameterization, see the following examples that use the FQDNs from Figure 6-15.

Example:

Script on the Linux agent **chm-linux-agent-d** (subsystem **D**):

```
chm_node_setup \
--subsys D \
--parent-subsys B \
--parent-chm chm-satellite-b
```

Example:

Script on the Windows agent **chm-win-agent-f** (subsystem **F**):

```
"C:\Program Files\chm\chm-node-setup.bat" \
--subsys F \
--parent-subsys C \
--parent-chm chm-satellite-c
```

6.10.4 Configuring multi-level HA monitoring

For this advanced usage scenario, you combine multi-level and high availability monitoring. This combination lets you realize, for example, a primary master that synchronizes all information with a distant secondary master. This usage scenario combines the features from the "pure" multi-level or HA monitoring configurations.

The following figure shows an example for such a multi-level, HA monitoring configuration.

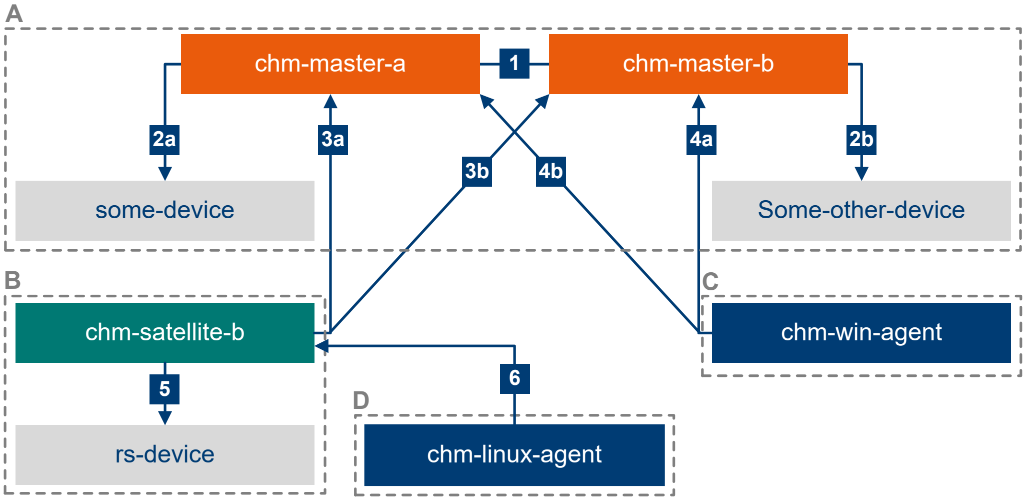


Figure 6-16: Multi-level, HA monitoring example

1 = Synchronization of monitoring results (HA configuration).
2a, 2b = Masters monitor devices.
3a, 3b = Satellite sends monitoring results to masters.
4a, 4b = Agent sends monitoring results to masters.
5 = Satellite monitors device.
6 = Agent sends monitoring results to satellite.

The top-level subsystem **A** comprises a primary master and a secondary master. Each of them directly monitors a device. The subsystems **C** and **D** comprise two agents. The agent in **D** is indirectly connected to the masters by the satellite in subsystem **B**. This satellite forwards monitoring results from the agent and directly monitors another device. The other agent in subsystem **C** is directly connected to both masters.

To set up a multi-level HA monitoring system

1. On all masters and satellites, install the R&S CHM host software.
   How to: Section 4, "Installing R&S CHM", on page 21
2. On each other host that masters or satellites cannot monitor with external checks, install the agent software.
   How to: Section 4.2, "Installing R&S CHM agents", on page 23
3. Install certificates and keys.
   How to: Section 6.10.5, "Deploying certificates for distributed monitoring", on page 92
4. Edit the `chm.yaml` file to describe the multi-level HA monitoring architecture.
   How to: Example "YAML configuration: multi-level HA monitoring" on page 90.
5. All masters and satellites require an identical `chm.yaml` file. Save this file here:
   `/etc/opt/rohde-schwarz/chm/`
6. Restart the chm service on all masters and satellites to take the changes effect:
   ```
   # systemctl restart chm
   ```
   We recommend starting the service in sequence on the masters and then on the satellite.
7. On each R&S CHM agent, run the node setup scripts with options that describe the multi-level HA system. See Section 6.10.4.2, "Configuring agents for multi-level HA monitoring", on page 91.

6.10.4.1 Editing the YAML configuration for multi-level HA monitoring

Multi-level HA monitoring configurations require that you configure subsystems for multi-level support and two masters for high-availability support.

► Specify the entries in the `chm.yaml` file for the multi-level HA monitoring configuration:

- The names of all subsystems
- The members of the subsystems, i.e. masters, satellites or agents, or monitored hosts or devices
- Exactly one parent subsystem except for the top-level subsystem
- Two R&S CHM host instances that serve as HA masters

Example: YAML configuration: multi-level HA monitoring

A satellite always requires a R&S CHM host installation on Linux. Thus, the satellite host requires the `connections: ["icinga2_linux"]` key.

This example:

- Uses the host names from Figure 6-16
- Omits any checks for clarity

```
subsystems:
  - name: "A"
    hosts:
      - "chm-master-a"
      - "chm-master-b"
      - "some-device"
      - "some-other-device"

  - name: "B"
    hosts:
      - "chm-satellite-b"
      - "rs-device"
    parent_subsystem: "A"

  - name: "C"
    hosts:
      - "chm-win-agent"
    parent_subsystem: "A"

  - name: "D"
    hosts:
      - "chm-linux-agent"
    parent_subsystem: "B"


hosts:
  # primary master in A
  - name: "chm-master-a"
    tags: ["chm"]

  # secondary master in A
  - name: "chm-master-b"
    tags: ["icinga2_ha"]

  # satellite in B
  - name: "chm-satellite-b"
    connections: ["icinga2_linux"]

  # windows agent in C
  - name: "chm-win-agent"
    connections: ["icinga2_win"]
```

```
# linux agent in D
- name: "chm-linux-agent"
  connections: ["icinga2_linux"]

# devices
- name: "some-device"
  checked_by: "chm-master-a"

- name: "some-other-device"
  checked_by: "chm-master-b"

- name: "rs-device"
```

The `checked_by` key for the host `some-other-device` ensures that this host is monitored by a specific R&S CHM instance, here the secondary master.

6.10.4.2 Configuring agents for multi-level HA monitoring

It is necessary that you inform the agents about these relations:

- The own subsystem.
- The parent subsystem.
- The connection to masters or satellites.
- The existence of both masters.

To configure the agents

► Run these scripts to complete agent configuration:

  - On Linux agents, run the `chm_node_setup` shell script.
  - On Windows agents, run the `chm-node-setup.bat` batch script.

  For parameterization, see the following examples that use the FQDNs from Figure 6-16.

Example:

Script on the Linux agent **chm-linux-agent-d** (subsystem **D**):

```
chm_node_setup \
--subsys D \
--parent-subsys B \
--parent-chm chm-satellite-b
```

Example:

Script on the Windows agent **chm-win-agent-f** (subsystem **C**):

```
"C:\Program Files\chm\chm-node-setup.bat" \
--subsys C \
--parent-subsys A \
--parent-chm chm-master-a
--second-parent-chm chm-master-b
```

6.10.5 Deploying certificates for distributed monitoring

If you configure high availability or multi-level monitoring, you currently have to provide your own certificate authority (CA) as described in Section 5.2, "Using CA-signed certificates", on page 35.

Add the following for every R&S CHM instance, i.e. master, satellite and agent, to the directories listed in Section 5.2, "Using CA-signed certificates", on page 35:

- A copy of the root certificate.
- Its own certificate signed by the CA.
- Its own private key corresponding to the signed certificate.

6.11 Using common keys

You can use the following common keys with any status check that is listed in Section 7, "Configuring status checks", on page 99.

checkgroups (Checkgroups)

Assigns a check to one or more specific groups that you can configure and display on the web GUI.

Example:

```
checkgroups: [Cluster, Buster]
```

Example: If the check group contains a colon (:), enclose the whole check group string in quotation marks.

```
checkgroups: ["Resources :- Disk space"]
```

displayname (Display name)

Display a user-friendly name on the web GUI.

Example:

```
displayname: My special service name
```

health_host (Check redirection)

FQDN of the host that provides status information for the SNMP-connected system component, e.g. a NAVICS or R&S RAMON device.

Use this key if you cannot obtain the status information from the system component itself but from a configured, central monitoring host.

Example:

```
health_host: navics_server.local
```

For an example in combination with the navics status check, see navics on page 124.

interval (Configure execution interval)

Configure an individual execution interval for a status check (in s, default: 60 s).

Example:

```
- idrac:
    snmp_connection:
      community: public
    interval: 30
```

logic_id (Logic identifier)

Assign a unique identifier to a check. You can specify this identifier in logic on page 48.

Ensure that all logic_id values are unique in the chm.yaml file.

Example:

```
checks
  - icinga2_cluster:
      logic_id: component1
  - dhcp
      logic_id: component2
  - dns
      logic_id: component3
```

maps (Coordinates for status icons on maps)

Specifies the coordinates for status icons on the maps.

Related parameters

Graphical system view (maps) on page 74

Parameters:

<map_name>	Name of the map as specified in Graphical system view (maps) on page 74.
x	The x-value on the image (horizontal, left to right).
y	The y-value on the image (vertical, up and down).
label_<format>	Item-specific label background, border or style. For more information about these keys, see Graphical system view (maps) on page 74.

Example: In these `host` and service configurations, the names of the maps are `Overview`, `Rack` and `Redundancy`. Compare with the example in `Graphical system view (maps)` on page 74.

```
hosts:
  - name: chm2-staging-disa.rsint.net
    displayname: "CHM Master"
    connections: [icinga2_api]
    tags: [chm]
    maps:
      Overview:
        x: 235
        y: 270
      Rack:
        x: 60
        y: 170
      Redundancy:
        x: 80
        y: 215
        label_background: "#AAAAAA"
# [...] some other keys
    checks:
      - icinga2_cluster:
          displayname: Icinga2 connect. via JSON/RPC on 5665/tcp
      - dhcp:
          maps:
            Overview:
               x: 250
               y: 208
            Rack:
              x: 60
              y: 300
              label_border: "#1E90FF"
            Redundancy:
              x: 620
              y: 100
```

6.12 Using frequent keys

You can use the following keys in multiple status checks. For example, you need SNMP in all checks that are based on this protocol, e.g. `nport` on page 126.

snmp_connection (SNMP connection)

Specifies the properties of the SNMP connection for communication between R&S CHM and the device.

- SNMPv1/v2: unencrypted communication
- SNMPv3: encrypted communication

An individual `snmp_connection` check overrules the `snmp_connection` host configuration.

Parameters:

port

numeric

Communication port at the device, the SNMP agent (optional).

*RST: 161

version

1 | 2 | 3

SNMP protocol version.

*RST: 2

retries

numeric

Number of retries to be used in the requests (optional).

*RST: 5

timeout

numeric

Timeout between retries (optional). Floating point numbers can be used to specify fractions of seconds, e.g. `1.25`.

*RST: 1
Default unit: s

community

string

SNMP community string for SNMPv1/v2 transactions. The community string is a type of shared password between the SNMP management station and the device. The community string is used to authenticate the SNMP management station.

*RST: public

trapreceivers

Configures R&S CHM to inform a list of SNMP notification receivers about system status changes via SNMPv2 traps (optional).

host
The host name of the SNMP notification receivers.

port
The port of the host.

community
The SNMP community that is expected by the SNMP notification receiver to accept the trap.

How to: Section 6.9.2, "Configuring SNMPv2 traps", on page 77 secname

string

Identifier (security name) used for authenticated SNMPv3 messages.
See also: Section 6.6, "Managing password identifiers", on page 66

authproto

MD5 | SHA | SHA-224 | SHA-256 | SHA-384 | SHA-512 | None

The authentication protocol that is used for authenticated SNMPv3 messages. If your operating system is hardened with FIPS mode, you cannot use `MD5`.

*RST: MD5

authpass

string

Password used for authenticated SNMPv3 messages (optional).
If not specified, R&S CHM looks up the password in the password store using the `secname` value as the identifier.
Option 1: Clear text password as used in the example at the end of this key description.
Option 2: VAULT:<path_to_vault> as used in the example at the end of this key description (recommended).
Option 3: If not specified, R&S CHM looks up the password in the password store using the `secname` value as the identifier as used in the example at the end of this key description.

privproto

DES | 3DES | AES-128 | AES-192 | AES-256 | None

Privacy protocol used for encrypted SNMPv3 messages.

*RST: DES

privpass

string

Password used for encrypted SNMPv3 messages (optional).
Option 1: Clear text password as used in the example at the end of this key description.
Option 2: VAULT:<path_to_vault> as used in the example at the end of this key description (recommended).
Option 3: If not specified, R&S CHM looks up the password in the password store using the `secname` value as the identifier as used in the example at the end of this key description.

context

string

Context name used for SNMPv3 messages, e.g. `spectracom_time`

*RST: empty string ""

seclevel

noAuthNoPriv | authNoPriv | authPriv

Security level used for SNMPv3 messages.

noAuthNoPriv
Authenticates with a username, i.e. no authentication and no encryption.

authNoPriv
Provides HMAC MD5 or SHA algorithms for authentication but no encryption.

authPriv
Provides HMAC MD5 or SHA algorithms for authentication and DES 56-bit encryption.

Example: SNMPv1/2

```
snmp_connection:
  port: 161
  version: 2
  community: public
```

Example: **SNMP v3, option 1**: Use the password store for a Spectracom SecureSync time server and write the passwords in clear text to the `chm.yaml` configuration file:

```
- spectracom_time:
    checkgroups: [water, earth, fire, air]
    snmp_connection:
      port: 1234
      version: 3
      secname: rsadmin
      authproto: SHA
      authpass: privatusprivatusprivatusprivatus
      # clear text password
      privproto: AES-256
      privpass: privatusprivatusprivatusprivatus
      # clear text password
      context: spectracom_time
```

Example: **SNMP v3, option 2**: Use the password store with different passwords for authpass and `privpass`:

```
- nport
    checkgroups: [water, earth, fire, air]
    snmp_connection:
      version: 3
      context: nport
      secname: mydeviceaccount  # the snmp user
      authpass: VAULT:snmp_passwords/nport/device1
      # The path to the password in the password store
      privproto: AES-256
      privpass: VAULT:snmp_passwords/nport/device1/privpass
      authproto: SHA
...
```

Example: **SNMP v3, option 3** (deprecated): Use the password store with identical passwords:

```
- nport:
    checkgroups: [water, earth, fire, air]
    snmp_connection:
      version: 3
      context: nport
      secname: mydeviceaccount
         # lookup of passwords in password store
      authproto: SHA
      privproto: AES-256
```

thresholds (Thresholds)

Specify thresholds for alert levels. Use `thresholds` together with suitable checks as mentioned in the description of the checks.

Thresholds are implemented according to the Monitoring Plugins Development Guidelines. The Table 6-8 is adopted from this guide.

Parameters:

warning Threshold for the `warning` alert level.

critical Threshold for the `critical` alert level.

Example:

```
thresholds:
  warning: ':0'   # E.g. alert if 1 or more exceed. occurred
  critical: ':0'  # E.g. alert if 1 or more exceed. occurred
thresholds:
  warning: '20:'  # E.g. alert if check cond. falls below 20
  critical: '10:' # E.g. alert if check cond. falls below 10
```

Generalized format of ranges:

`[@]start:end`

Table 6-8: Example ranges

Range definition	Generate an alert if x...
10	< 0 or > 10 (outside the range of {0 to10})
10:	< 10 (outside {10 to ∞})
~:10	> 10 (outside the range of {-∞ to 10})
10:20	< 10 or > 20 (outside the range of {10 to 20})
@10:20	≥ 10 and ≤ 20 (inside the range of {10 to 20})

7 Configuring status checks

R&S CHM provides a specific set of status checks that you can configure. Here, you can obtain an overview of available status checks and necessary information on how to configure them.

For common keys that are supported by all status checks, see Section 6.11, "Using common keys", on page 92.

Table 7-1: Syntax conventions

Identifier	Description
*RST	Default value

bitdefender (Bitdefender virus definitions age; deprecated)

Monitors the age of the virus definitions of Bitdefender antivirus software.

Related parameters

- `thresholds`

Parameters:

thresholds

warning | critical

Alert levels for the age of the definition base (in days).
For more information about the `thresholds` syntax, see `thresholds` on page 98.

Example:

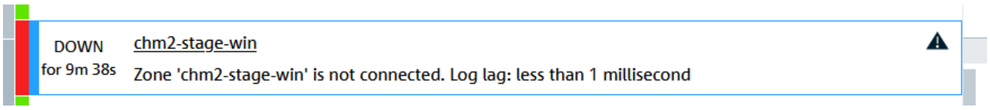
```
checks:
- bitdefender:
  thresholds:
    warning: '10'
    critical: '30'
```

chm_agent_connection (CHM agent connection)

Checks the connection between the R&S CHM host and the R&S CHM service that runs on an agent. This check enhances the reliability of the returned status.

Return status values for checked agents:

- "UP" if the service is running and connection is possible.
- "DOWN" if the service is not running or connection is not possible.



DOWN
for 9m 38s

chm2-stage-win

Zone 'chm2-stage-win' is not connected. Log lag: less than 1 millisecond

You can configure this check for agents instead of `ping`.

Example:

```
checks:
  - chm_agent_connection:
```

chm_remote, simcos3 (RS-RAMON-CHM-REMOTE connection)

Monitors any device that implements RS-RAMON-CHM-REMOTE MIB, e.g.
R&S RAMON and R&S SIMCOS.

Related parameters

- snmp_connection

Parameters:

appid
string
The identifier of the software, see Table 7-3.

checkid
string
The identifier of the device, see Table 7-3.
With R&S SIMCOS, set the checkid that you have specified during device configuration.
With R&S SIMCOS, set the checkid that you have specified during device configuration.

Example:
Alternative 1

```
- chm_remote:
    snmp_connection:
      port: 1234
      version: 2
      community: public
    appid: SIMCOSIII
    checkid: MODEM 1
```

Example:
Alternative 2

```
- simcos3:
    snmp_connection:
      port: 1234
      version: 2
      community: public
    checkid: MODEM 1
```

chm_remote_grpc (gRPC-based RAMON monitoring)

Monitors health summary and status of R&S RAMON and R&S SIMCOS. For concepts a configuration instruction, see Section 6.7, "Configuring R&S RAMON for monitoring", on page 68.

Parameters:

appid
string
The identifier of the software, see Table 7-3.

checkid
string
The identifier of the device, see Table 7-3.
With R&S SIMCOS, set the checkid that you have specified during device configuration.

port

numeric

Remote TCP port.

*RST: 18005

server_root_cert

string

Path of the file that contains the PEM encoded root certificate of the target host. The certificate is used for authenticating the target host.

*RST: /var/lib/icinga2/certs/ca.crt

client_root_cert

string

Path of the file that contains the PEM encoded root certificate of the local host. The certificate is used by the server in combination with `client_cert` for authenticating the local host.

*RST: /var/lib/icinga2/certs/ca.crt

client_cert

string

Path of the file that contains the PEM encoded certificate of the local host. The certificate is used by the server in combination with `client_root_cert` for authenticating the local host.

*RST: /var/lib/icinga2/certs/<localhost_fqdn>.crt

client_privkey

string

Path of the file that contains the PEM encoded private key that corresponds to `client_cert` of the local host.

*RST: /var/lib/icinga2/certs/<localhost_fqdn>.crt

insecure

boolean

If set to true, try connecting without encryption and client/server authentication.

*RST: false

Example:

```
hosts:
  - name: applicationserver.some.net
    checks:
      - chm_remote_grpc:
          appid: SIMCOSIII
          checkid: 1
          server_root_cert: /var/certs/srv_ca.crt
          client_root_cert: /var/certs/cl_ca.crt
          client_cert: /var/certs/cl.crt
          client_privkey: /var/keys/cl.key
```

Table 7-2: Supported software and identifiers, only for - chm_remote_grpc

Software	appid	checkid
R&S EWCoM	EWCoMApplication	EWCoMHealthCheck1

Table 7-3: Supported software and identifiers for - chm_remote_grpc, - chm_remote, - simcos3

Software	appid (<x> is the number of the device)	checkid
R&S SIMCOS	SIMCOSIII	<checkid>
R&S RAMON CA120	CA120Server	StorageUnits
R&S RAMON CA120	CA120Server	ProcessingUnits
R&S RAMON CA120	CA120Server	Tuners
R&S RAMON CA120	CA120Server	Server
R&S RAMON Antennamatrix	AntennaMatrixDRV1 to AntennaMatrixDRV<x>	ChmSnmpCheck1
R&S RAMON Amrec	AMRECServer1 to AMRECServer<x>	AMRECDevices
R&S RAMON Driver DDF007	DDF007DRV1 to DDF007DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver DDF1555	DDF1555DRV1 to DDF1555DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver DDF200M	DDF200MDRV1 to DDF200MDRV<x>	RxChmSnmpCheck1
R&S RAMON Driver DDF205	DDF205DRV1 to DDF205DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver DDF255	DDF255DRV1 to DDF255DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver DDF260	DDF260DRV1 to DDF260DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver DDFCTL	DDFCTLDRV1 to DDFCTLDRV<x>	RxChmSnmpCheck1
R&S RAMON Driver WPU500	WPUCTLDRV1 to WPUCTLDRV<x>	RxChmSnmpCheck1
R&S RAMON Driver EM100	EM100DRV1 to EM100DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver ESMD	ESMDDRV1 to ESMDDRV<x>	RxChmSnmpCheck1
R&S RAMON Driver ESME	ESMEDRV1 to ESMEDRV<x>	RxChmSnmpCheck1
R&S RAMON Driver ESMW	ESMWDRV[1] to ESMWDRV[x]	RxChmSnmpCheck1
R&S RAMON Driver EB200	EB200DRV1 to EB200DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver EB500	EB500DRV1 to EB500DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver EB510	EB510DRV1 to EB510DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver PR100	PR100DRV1 to PR100DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver PR200	PR200DRV1 to PR200DRV<x>	RxChmSnmpCheck1
R&S RAMON Driver EM200	EM200DRV1 to EM200DRV<x>	RxChmSnmpCheck1
R&S RAMON RACAS	RaCas	1
R&S RAMON SIGDB	SIGDB	1
R&S BBI	BBI	GenChk
R&S BBI	BBI	MemChk
R&S BBI	BBI	ConChk
R&S BBI	BBI	SigChk
R&S BBI	BBI	KeyCalcChk

Software	appid (<x> is the number of the device)	checkid
R&S BBO	BBO	GenChk
R&S BBO	BBO	MemChk
R&S BBO	BBO	ConChk
R&S BBO	BBO	DevoChk
R&S DCU	DCU	GenChk
R&S DCU	DCU	MemChk
R&S DCU	DCU	IfChk
R&S DCU	DCU	KeyCalcChk
R&S DCU	DCU	FPGAChk
R&S DCU	DCU	ProdChk
R&S GSA6Sensor	GSA6Sensor	GenChk
R&S GSA6Sensor	GSA6Sensor	MemChk
R&S GSA6Sensor	GSA6Sensor	HealthChk
R&S GSA6Sensor	GSA6Sensor	SigChk
R&S GSA6Sensor	GSA6Sensor	DbChk
R&S GSA6Sensor	GSA6Sensor	ProdChk
R&S Linkmanager	LnkMngr	GenChk
R&S Linkmanager	LnkMngr	MemChk
R&S Linkmanager	LnkMngr	HealthChk
R&S Linkmanager	LnkMngr	ConChk
R&S Linkmanager	LnkMngr	NtwrkChk
R&S Receiverserver	RcvSrv	GenChk
R&S Receiverserver	RcvSrv	MemChk
R&S Receiverserver	RcvSrv	HealthChk
R&S Receiverserver	RcvSrv	SigChk
R&S Receiverserver	RcvSrv	ConChk
R&S Receiverserver	RcvSrv	SynchChk
R&S Receiverserver	RcvSrv	ProdChk
R&S SBU	SBU-T	GenChk
R&S SBU	SBU-T	MemChk
R&S SBU	SBU-T	SigChk
R&S SBU	SBU-T	ConChk
R&S SBU	SBU-T	SynchChk

Software	appid (<x> is the number of the device)	checkid
R&S SBU	SBU-T	ProdChk
R&S SCG	SCG	GenChk
R&S SCG	SCG	MemChk
R&S SCG	SCG	HealthChk
R&S SCG	SCG	ConChk
R&S SCG	SCG	QualChk
R&S SCM	SCM	GenChk
R&S SCM	SCM	MemChk
R&S SCM	SCM	DbChk
R&S SCM	SCM	ConChk
R&S Sensorserver	SNS	GenChk
R&S Sensorserver	SNS	MemChk
R&S Sensorserver	SNS	ConChk
R&S Sensorserver	SNS	ShrdFldChk
R&S Sensorserver	SNS	ProdChk

cisco_hardware (Cisco hardware)

cisco_hardware.py

Monitors the hardware status of a Cisco switch via SNMP. The check monitors fans, temperature, power supplies and modules.

Supported devices

All devices that support the following MIBs, including Cisco Catalyst 9300:

- CISCO-ENVMON-MIB
- CISCO-STACKWISE-MIB
- CISCO-ENTITY-FRU-CONTROL-MIB

Related parameters

- `snmp_connection`

Parameters:

device_name string

Name of the device. This name is shown in the status summary (optional).

return_status CRITICAL | WARNING

Return status for failures (optional).

fans
numeric
Number of built-in fans (optional).
*RST: 2

powersupplies
numeric
Number of built-in power supplies (optional).
*RST: 2

Example:

```
- cisco_hardware:
    device_name: CISCO 9300 Center Switch
    fans: 3
    returnstatus: WARNING
```

---

cputemp (Monitor average CPU temperature)

Monitors the CPU package temperature for all CPUs on a Windows host. The CPU package temperature is a 256 millisecond average value of the hottest temperature sensor.

Related parameters

- `thresholds`

Parameters:

thresholds
warning | critical
Check-specific alert levels. For more information about the threshold syntax, see `thresholds` on page 98 (optional).
*RST: warning: 80, critical: 90
Default unit: °C

Example:

```
- cputemp:
    thresholds:
      warning: '70'
      critical: '90'
```

---

dhcp (DHCP server)

Tests the availability of DHCP servers on a network. By default, the check broadcasts a DHCPDISCOVER packet to port 67/UDP and checks whether a DHCPOFFER is received on 68/UDP within a given timeout.

Related parameters

- `thresholds`

Parameters:

servers
IP_address1 , IP_address2 , IP_address<n>
List of IP address of DHCP servers from which an answer is expected (optional). If multiple servers are specified, and some but not all respond, this situation results in a warning alert.
*RST: Any responding DHCP server is ok

offeredip IP_address

Expected IP address in DHCPOFFER (optional). If specified, and a DHCPOFFER with unexpected IP is received, this situation results in a warning alert.

*RST: Any offered IP address is ok

timeout time

Time to wait for DHCPOFFER (optional).

*RST: 2
Default unit: s

interface string

Interface to be used for listening (optional).

*RST: eth0

mac string

MAC address to use in the DHCP request (optional).

*RST: MAC address of the configured interface

unicast true | false

If set to `true`, mimics a DHCP relay (optional). Requires to set also at least one server.

*RST: false

Example:

```
-dhcp
  servers: [192.168.178.0 , 192.168.178.1]
  unicast: true
```

dkn (Devices and nodes in a DKN)

R&S CHM lets you monitor the status of devices and nodes (BACs) by using the GEDIS KMS RLM SNMP MIB in a NEMAS DKN from the Thales Group.

Related parameters

- `snmp_connection`

For returned status values, see Table 7-4.

Parameters:

type device_ready | device_status | node_link | node_status

Check type.

device_ready
Monitor if a DKN device is in ready state.

device_status
Monitor the status of a DKN device.

node_link
Monitor the link status of the node.

node_status
Monitor the node status.

id numeric

Identifier of the DKN device or node.

Example:

```
- dkn:
  snmp_connection:
    version: 2
    community: public
  type: device_ready
  id: 1
- dkn:
  snmp_connection:
    version: 2
    community: public
  type: device_status
  id: 1
- dkn:
  snmp_connection:
    version: 2
    community: public
  type: node_link
  id: 2
- dkn:
  snmp_connection:
    version: 2
    community: public
  type: node_status
  id: 2
```

Table 7-4: Status mapping - DKN to web GUI

Check type	Status on web GUI	DKN status
node_link	🟩 "OK"	Connected
	🟥 "CRITICAL"	Disconnected
device_ready	🟩 "OK"	Ready
	🟥 "CRITICAL"	Not ready
device_status, node_status	🟩 "OK"	OK, Info
	🟧 "WARNING"	Warning
	🟥 "CRITICAL"	Error, Fatal

dns (DNS server)

Tests the availability of DNS servers on a network. The default servers from `/etc/resolv.conf` are used unless explicitly specified.

Related parameters

- `thresholds`

Parameters:

lookup

string

The host name or IP to query the DNS for (optional).

*RST: Name of host where check is executed

server

IP_address

The DNS server to query.

*RST: The server configured in the OS.

query_type

A | AAAA | SRV | TXT | MX | ANY

The DNS record type (optional).

A
IPv4 address record.

AAAA
IPv6 address record.

SRV
Service location record.

TXT
Text record.

MX
Mail exchange record.

ANY
A special query (meta-query, deprecated).

*RST: A

answers

string

The answers to look for. A host name must end with a dot. Define multiple answers as array (optional).

*RST: Do not check for specific addresses in answer

authoritative

true | false

Expect the server to send an authoritative answer. Non-authoritative answers are marked with "non-authoritative answer:" and mean that a name server looked up the entry from it is local cache (optional). If set to false, there is no check whether authoritative or not.

*RST: false

accept_cname

Accept CNAME (canonical name, aka alias) responses as a valid result to a query (optional).

timeout

numeric

Seconds before connection times out, i.e. forced interruption by SIGALRM, then SIGKILL (optional).

*RST: 10
Default unit: s

thresholds

Alert levels for used datastore space (optional).
For more information about the `thresholds` syntax, see `thresholds` on page 98.

Example:

```
- dns
  lookup: my_dnsserver
  accept_cname:
  timeout: 20
```

domain (Monitor a domain)

Monitors a domain using the given check type.

Parameters:

type

sec_channel | replication | membership

The type of domain-check to create.

sec_channel
Tests the secure channel between the local computer and its domain.

replication
Tests the replication between the domain controllers.

membership
Checks the Windows domain membership of a Windows host.

domainName

string

The domain to monitor (only optional for `replication`).

Example:

```
domain:
  type: sec_channel
  domainName: rsint.net
```

Example:

```
domain:
  type: replication
```

Example:

```
domain:
  type: membership
  domainName: rsint.net
```

dummy (Dummy)

The check always shows the status "UP" for the host. Use this check if you cannot use another host check, e.g. if ICMP is blocked in the network.

Example:

```
- name: host_prepare.net
  checks:
    - dummy:
```

eta_pdu (Monitor ETA PDUs)

Monitors PDUs from ETA that support the MIB `eta_RCI11_1.0.1_MIB.mib`.

Monitored aspects:

- Serial number of the fuse (info only)
- Status of fuse
- No system parameter available fault
- Parameter CRC fault
- Program memory CRC fault
- Internal memory fault
- Controller fault
- Watchdog reset fault
- Output status of the fuse
- Short is detected by this fuse
- Overload status
- Undervoltage status
- Overvoltage status
- Overtemperature status

Related parameters

- `thresholds`

Parameters:

fuse_number numeric

The fuse number to check (optional).

fuse_feed A | B

The fuse feed to check (`A` or `B`; optional).

temperature_sensor_id numeric

The ID of the temperature sensor to check (optional).

humidity_sensor_id numeric

The ID of the humidity sensor to check (optional).

Example: Check fuse no. `3` and fuse feed `B`.

```
- eta_pdu:
    fuse_number: 3
    fuse_feed: B
```

Example: Check fuse no. `1`, fuse feed `A`, temperature sensor ID and humidity sensor ID.

```
- eta_pdu:
    fuse_number: 1
    fuse_feed: "A"
- eta_pdu:
    temperature_sensor_id: 1
    thresholds:
      temperature:
        warning: ":30"
        critical: ":40"
- eta_pdu:
    humidity_sensor_id: 1
    thresholds:
      humidity:
        warning: ":80"
        critical: ":90"
```

file_content (Monitor file content)

Monitors the content of a file for a predefined string on Linux and Windows agents.

Parameters:

file
string
Name of the monitored file (optional).
*RST: /tmp/import_service_result

string
string
Search string. Mandatory on Linux agents and not applicable on Windows agents.

pattern
string
Search string, expressed as a regular expression in .Net syntax. Mandatory on Windows agents and not applicable on Linux agents.

match_is_ok
true | false
If `true`, a match is interpreted as ok (default). If `false`, a match is interpreted as failure.

returnstatus
WARNING | CRITICAL
Return value if the check fails, i.e. "WARNING" or "CRITICAL" (optional).

oksummary
string
Text that is shown if the string is found in the file.

badsummary
string
Text that is shown if the string is not found in the file.

showcontent string

Content of the file in the long output.

Example: Example of a Linux agent that uses a search string.

```
- file_content:
    file: /tmp/import_service_result
    string: specific_search_string
    returnstatus: CRITICAL
    oksummary: Import Service OK
    badsummary: Import Service FAILED
```

Example: Example of a Windows agent that uses a search pattern.

```
- file_content:
    file: C:\Users\Operator\log.txt
    match_is_ok: false
    returnstatus: warning
    pattern: ^\s.*\d{10}.+abc.*\{\|\}~$
```

fortinet (Fortinet controller)

Monitors the status of a controller from Fortinet Inc.

Related parameters

- snmp_connection

Parameters:

resources true

Check the controller resources (optional).

controller true

Check the controller status (optional).

accesspoints true

Check the access points (optional).

Example: Monitor Fortinet controllers and access points.

```
- fortinet:
    snmp_connection:
      version: 2
      community: fortinet_ok
      port: 1234
    resources: true
    controller: true
    accesspoints: true
```

Example: Monitor Fortinet access points.

```
- fortinet:
    snmp_connection:
      version: 2
      community: fortinet_nok
      port: 1234
    accesspoints: true
```

gb2pp (gb2pp server check over an R&S trusted filter)

Queries gb2pp servers for system or host group summary states to transfer these data via an R&S TF5900M trusted filter IP.

For details about the state aggregation logic, see Section 6.2, "Understanding aggregated states", on page 39.

The following figure shows how the status check works.

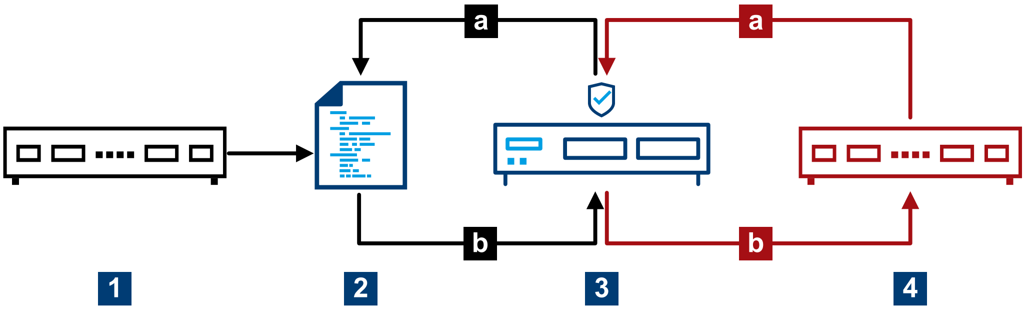
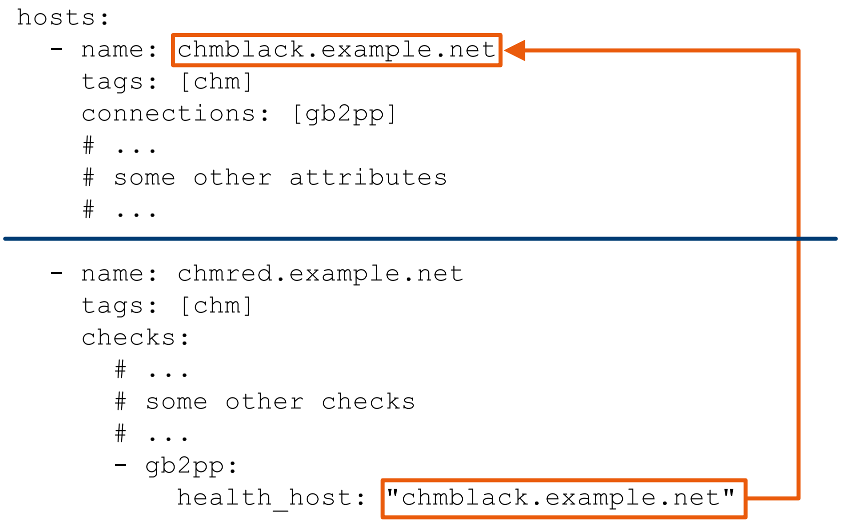


Figure 7-1: Conceptual representation of the gb2pp service check

1 = Host name: chmblack.example.net
2 = Monitoring data (gb2pp format)
3 = R&S TF5900M trusted filter IP
4 = Host name: chmred.example.net
a = Request monitoring data
b = Response

The following figure illustrates the relationship between the `health_host` key and the host name, i.e. the name of the gb2pp server.



```
hosts:
  - name: chmblack.example.net
    tags: [chm]
    connections: [gb2pp]
    # ...
    # some other attributes
    # ...

  - name: chmred.example.net
    tags: [chm]
    checks:
      # ...
      # some other checks
      # ...
      - gb2pp:
          health_host: "chmblack.example.net"
```

Figure 7-2: Relation between involved keys

The `gb2pp` check only works in combination with `hosts` on page 41 > `connections: ["gb2pp"]`.

Trusted filter devices between gb2pp server and client can possibly block TCP packets that contain TCP time stamps.

If so, disable TCP time stamps as follows:

- Run this command: `sysctl -w net.ipv4.tcp_timestamps=0`
- Add the line `net.ipv4.tcp_timestamps=0` to the default `sysctl.conf` file. You can find this file here: `/etc/sysctl.conf`.

Parameters:

health_host

server_name

Checks the system state of this gb2pp server. Specify the `name` of that host.

hostgroup

string

Checks the summary state of a host group (optional). Only in combination with `health_host`.

Example:

System state check

```
hosts:
  - name: chmblack.example.net
    tags: [chm]
    connections: [gb2pp]
    # ...
    # some other attributes
    # ...

  - name: chmred.example.net
    tags: [chm]
    checks:
      # ...
      # some other checks
      # ...
      - gb2pp:
          health_host: "chmblack.example.net"
```

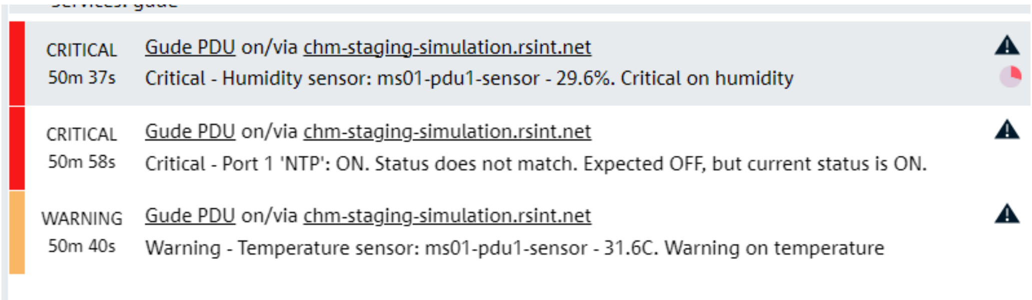
Example: **Host group state check**

```
hosts:
  - name: "chmblack.example.net"
    tags: ["chm"]
    connections: ["gb2pp"]
    hostgroups: ["saturn"]
    # ...
    # some other attributes
    # ...

  - name: "chmred.example.net"
    tags: ["chm"]
    checks:
      # ...
      # some other checks
      # ...
      - gb2pp:
          health_host: "chmblack.example.net"
          hostgroup: "saturn"
```

gude (Monitor a Gude PDU)

Monitors temperature, humidity sensor and outlets of a Gude power distribution unit (PDU), e.g. Gude 8045.



Services: gude

CRITICAL 50m 37s — Gude PDU on/via chm-staging-simulation.rsint.net
Critical - Humidity sensor: ms01-pdu1-sensor - 29.6%. Critical on humidity

CRITICAL 50m 58s — Gude PDU on/via chm-staging-simulation.rsint.net
Critical - Port 1 'NTP': ON. Status does not match. Expected OFF, but current status is ON.

WARNING 50m 40s — Gude PDU on/via chm-staging-simulation.rsint.net
Warning - Temperature sensor: ms01-pdu1-sensor - 31.6C. Warning on temperature

Figure 7-3: Example check results on the R&S CHM web GUI

Related parameters

- snmp_connection
- thresholds

Parameters:

check_temperature true | false
Enables checking the temperature sensor (optional).

check_humidity true | false
Enables checking the humidity sensor (optional).

port_number numeric
Specifies the port number to check status for (optional).

expected_status ON | OFF

The expected status for the port (optional).

Example: Check the temperature.

```
- gude:
    snmp_connection:
      version: 2
      community: gude
    check_temperature: true
    thresholds:
      temperature:
        warning: ':30'
        critical: ':40'
```

Example: Check the humidity.

```
- gude:
    snmp_connection:
      version: 2
      community: gude
    check_humidity: true
    thresholds:
      humidity:
        warning: ':80'
        critical: ':90'
```

Example: Check the port status.

```
- gude:
    snmp_connection:
      version: 2
      community: gude
    port_number: 1
    expected_status: "OFF"
```

hums (CHM instrument health & utilization)

Checks health and utilization data of R&S CHM instruments via LXI.

Example:

```
- hums:
```

icinga2_cluster (Icinga2 cluster)

Checks if all endpoints in the current Icinga2 zone and the directly connected zones are working properly.

Example:

```
- icinga2_cluster:
    logic_id: component1
```

idrac (Dell iDRAC hardware)

Monitors the hardware status of a server with a Dell iDRAC interface via SNMP.

Checked values

- Global system status
- Global LCD status
- System power
- Global storage status
- Power unit redundancy
- Power unit status
- Chassis intrusion sensor status
- Cooling unit status
- Status of all drives
- Predictive status of all drives
- All temperatures

If a component does not exist or if a sensor in the server version does not exist, set this check manually to `true`. For example, if there are no hard disks (diskless server), set the key `no_disks` to `true`.

Related parameters

- `snmp_connection`

Parameters:

no_storage
true
Do not check global storage condition (optional).

no_system
true
Do not check global system status (optional).

no_power
true
Do not check global power status (optional).

no_temperature
true
Do not check overall thermal environment condition (optional).

no_disks
true
Do not check the disks (optional).

no_power_unit
true
Do not check the power unit (optional).

no_intrusion
true
Do not check the intrusion sensor (optional).

no_cooling
true
Do not check the cooling unit (optional).

no_redundancy
true
Do not check the power unit redundancy (optional).

no_predictive
true
Do not check the predictive status of the disks (optional).

no_lcd
true
Do not check the LCD status (optional).

Example:

```
- idrac:
    no_power_redundancy: true
```

ilo (HP iLo hardware)

Monitors the hardware status of a server with Hewlett-Packard iLO interface via SNMP.

Checked values

- Global storage status
- Global memory status
- Global system status
- Global power supply status
- Global power state (ON/OFF)
- Global thermal system
- Global temperature sensors
- Global fan status
- Disk controllers
- Power supply redundancy
- Fans
- Disk drive status
- Disk drives smart values
- Disk temperatures

If a component or a sensor does not exist, set this check manually to `true`.

Related parameters

- `snmp_connection`

Parameters:

drives
numeric
Number of physical drives.

ps
numeric
Number of connected power supplies.

fan
numeric
Number of fans.

[no_storage]
true
Do not check global storage condition (optional).

[no_system]
true
Do not check global system state (optional).

no_powersupply
true
Do not check global power supply condition (optional).

no_powerstate true

Do not check power state (optional).

no_temp true

Do not check overall thermal environment condition (optional).

no_temp_sensors true

Do not check temperature sensor condition (optional).

no_temp_drives true

Do not check the temperature sensor of the hard disk drives (optional).

no_fan true

Do not check global fan condition (optional).

no_memory true

Do not check memory condition (optional).

no_controller true

Do not check controller condition (optional).

no_logical_drives true

Do not check the logical drives (optional).

no_power_redund true

Do not check power supply redundancy (optional).

Example:

```
- ilo:
   drives: 2
   ps: 1
   fan: 3
   no_power_redund: true
```

lancom_vpn_status (Monitor VPN connection status)

Monitors the status of VPN connections on a LANCOM device via SNMP.

Related parameters

- `snmp_connection`

Parameters:

connection string

The name of the VPN connection to check.

Example:

```
checks:
  - snmp_hostalive:
  - lancom_vpn_status:
      connection: "PROJECT"
      snmp_connection:
        context: lancom_vpn_status
  - lancom_vpn_status:
      connection: "SECOND"
      snmp_connection:
        context: lancom_vpn_status
```

---

load (CPU load)

Monitors CPU load on Windows and Linux hosts.

Related parameters

- thresholds

Parameters:

thresholds

warning | critical

Check-specific alert levels. For more information about the threshold syntax, see thresholds on page 98. The following values only apply to the current load on Windows.
For Linux, see load<minutes>.

*RST: warning: 90, critical: 99
Default unit: %

load<minutes>

warning | critical

On Linux, check load averages in the last 1 min, 5 min and 15 min (fixed). The threshold defines the utilization ratio of all processor cores.
The Linux load averages depend on the number of processor cores. For a single-core processor, a load of 1.0 means that the processor is exactly at capacity. Smaller values indicate that there is still capacity available. Higher values indicate problems, i.e. the system is slowing down or hanging.
On a multicore system, ensure that the load does not exceed the number of cores available. It does not matter how the cores are spread out over CPUs. **Two quad-cores** match **four dual-cores** match **eight single-cores**, i.e. in sum consider **eight cores** when configuring the alert levels.

Increment: 0.01
Default unit: numeric
For alert level defaults, see Table 7-5.

Example:

On Windows

```
-load:
   thresholds:
     warning: '90'
     critical: '99'
```

Example:

On Linux

```
- load:
    thresholds:
      load1:
        warning: '5.0'
        critical: '10.0'
      load5:
        warning: '4.0'
        critical: '6.0'
      load15:
        warning: '3.0'
        critical: '4.0'
```

Table 7-5: Load threshold defaults on Linux

Load averaging	Alert level and threshold defaults
load1	warning: 5.0
	critical: 10.0
load5	warning: 4.0
	critical: 6.0
load15	warning: 3.0
	critical: 4.0

manual (Add GUI button "manual" ("manual" check))

Adds a button for "passive" checks on the web GUI. If there is a device in the system that cannot be monitored, you can enter the manual status here, using the "Process manual check result" button.

Figure 7-4: Web GUI example

To view the button on the web GUI, the user needs the permission `manual`. See `authorization` on page 63 > `permissions`.

Example:

```
checks:
  - ping:
  - manual:
      displayname: "<my_check_name>"
```

meinberg (Monitor Meinberg NTP)

Monitors the network time protocol (NTP) current state and GPS mode for devices that support the MBG-LANTIME-NG-MIB.

If the status is something else than "synchronized", R&S CHM returns a "WARNING" for the NTP current state and "CRITICAL" for the GPS mode.

Also, you can define a threshold for the good available satellites. E.g., if there are fewer than 5 satellites available, the status is "CRITICAL".

Related parameters

- `snmp_connection`
- `thresholds`

All keys are optional (`thresholds`, `satellites`, `warning` and `critical`).

Parameters:

satellites warning | critical

Defines the thresholds for the number of tracked satellites (optional).

*RST: warning: 5, critical: 3

Example: Monitoring configuration:

```
- meinberg:
    thresholds:
      satellites:
        warning: '10:'
        critical: '5:'
```

Example: Output on the R&S CHM web GUI:

```
"Critical - GPS Position: 48.1276 11.6124 619m.
Ntp Current State Int status: NOT_SYNCHRONIZED.
Gps Mode Int status: GPS_WARM_BOOT. Good satellitess: 7
```

---

mikrotik (MikroTik switches and router)

Monitors various aspects of a MikroTik device via SNMP.

Parameters:

check_power_supply1true | false

Checks power supply 1 status (optional).

check_power_supply2true | false

Checks power supply 2 status (optional).

check_fan1 true | false

Checks fan 1 speed (optional).

check_fan2 true | false

Checks fan 2 speed (optional).

check_hitemp true | false

Checks the HI temperature (optional). Not in combination with `check_devtemp`.

check_devtemp

true | false

Checks the DEV temperature (optional). Not in combination with `check_hitemp`.

temperature

warning | critical

If you configure a temperature check, specify the corresponding `thresholds` (optional).

Example:

```
- mikrotik:
   check_power_supply1: true
   check_power_supply2: true
   check_fan1: true
   check_fan2: true
   check_hitemp: true
   check_devtemp: false
   thresholds:
     temperature:
       warning: ":30"
       critical: ":40"
```

navics (Monitor NAVICS)

Monitors the status of an IP based naval communications system from Rohde & Schwarz (NAVICS).

Parameters:

type

server | groupserver | gw | cwp | sip | baa

Monitored NAVICS component. All components require an `equid` (equipment ID) or a `name`, except for the `groupserver`.

server
A session border control server.

groupserver
A radiotelephony control server.

gw
A media gateway.

cwp
A voice terminal.

sip
A SIP device.

baa
The NAVICS broadcast and alarm system (BAA).

master

string

If you configure `type: baa`, specify a `master` as the name of the first BAA media gateway.

agent

string

If you configure type: baa, specify an agent as the name of the secondary BAA media gateway.

equid

string

Equipment ID for type cwp and type sip.

name

string

Name for type gateway (gw) and type server (server).

Example:

```
- navics:
    type: server
    name: RADIO_SERVER
```

Example:

```
- navics:
    type: cwp
    eqid: EQID-VT-108
```

Example:

1) Typical NAVICS example

There is a host with a name navicsbaseserver.example.net and a service navics. The host is checked using ping and the service check is navics.

```
- name: navicsbaseserver.example.net
  connections: [snmp]
  snmp_connection:
    community: public
  checks:
    - ping:
    - navics:
        type: cwp
        eqid: VT1
```

This example has a disadvantage. You are monitoring voice terminals (VTs) but you cannot get the status information directly from the VTs. Instead, you ask the NAVICS server. To show the all the monitored VTs on the web GUI, you must specify them under the NAVICS server:

```
host: navicsbaseserver.example.net
   - service 1: Voice Terminal 1 status
   - service 2: Voice Terminal 2 status
   - service 3: Voice Terminal 3 status
   - service 4: Voice Terminal 4 status
   - ...
   - service 199: Voice Terminal 199 status
```

On the web GUI, all voice terminal status values are then also listed below the NAVICS server.

Example:

2) NAVICS example using the `health_host` key

To increase the overview of voice terminals on the web GUI, you can show every instance as a single host. If you can ping the voice terminals directly, you ask the NAVICS server for status information. Of course, you also configure the NAVICS server in the `chm.yaml` file.

```
- name: VT1.navics
    connections: [snmp]
    checks:
      - ping:
      - navics:
          health_host: navicsbaseserver.example.net
          type: cwp
          eqid: VT1

- name: navicsbaseserver.example.net
    connections: [snmp]
    snmp_connection:
      community: public
    checks:
      - ping:
```

If you cannot ping the voice terminals, you can configure a logic function as described in `logic` on page 48. Here, you also find a detailed NAVICS configuration example.

See also: `health_host` on page 92

Example:

3) Monitoring of the NAVICS broadcast and alarm system

```
- navics:
    type: baa
    master: ZYNQ_RGW3
    agent: ZYNQ_RGW4
    health_host: chm-staging-simulation.rsint.net
    snmp_connection:
      context: navics
```

nport (Moxa NPort 6000 series server)

Monitors a Moxa NPort 6000 series serial server via SNMP.

Supported MIBs

- RFC1213-MIB
- MOXA-NP6000-MIB

Related parameters

- `snmp_connection`

Parameters:

serial_port numeric

Monitored serial port.

name	string

Port name.

errormessage	string

Additional error message that indicates the status failure.

returnstatus	"CRITICAL" | "WARNING"

Returns status for failures.

dsr , cts , dtr	HIGH | LOW

Checks for the serial DSR, CTS or DTR flow control if the OK status is HIGH or LOW.

counter	Checks the port for frame, break, overrun and parity error counters (optional).

Example:

```
-nport:
   serial_port: 2
   dtr: LOW
   dsr: HIGH
   cts: LOW
   errormessage: "GENERATOR FAILED"
   name: "GENERATOR INPUT"
   returnstatus: "WARNING"
   counter:
-nport:
   serial_port: 3
   dtr: LOW
   errormessage: "AIRCONDITION FAILED"
   counter:
```

ntp_time (NTP server time synchronization)

Monitors time synchronization with an NTP server running on Windows or Linux. Only UTC time is used for calculating time offsets between client and server, even if your NTP client or server uses other timezones to display daytime.

Related parameters

- thresholds

Parameters:

server	FQDN | IP_address

FQDN, IPv4 or IPv6 address of the NTP server.

port	numeric

NTP port of the server (optional).

*RST:	123

timeout time

Seconds before connection times out (optional).

*RST: 10
Default unit: s

offset time

Expected time offset in seconds. Thresholds get adjusted automatically (optional).

*RST: 0
Default unit: s

thresholds warning | critical

Alert levels for time offset to NTP server (optional).
For more information about the `thresholds` syntax, see `thresholds` on page 98.

*RST: warning: '-0.1:0.1', critical: '-0.5:0.5'
Default unit: s

Example:

```
- ntp_time:
    server: ntpserver.example.com
    port: 12345
    timeout: 5
    offset: 3600
    thresholds:
      warning: '-0.5:0.5'
      critical: '-1:1'
```

nw_interface (Network interface)

Monitors the status of the network interface of devices that implement the RFC1213-MIB via SNMP.

Checked values

- Speed of the network interface
- Operational status
- Administrative status
- Port security MAC based
- Port security 702.1x based

Related parameters

- `snmp_connection`

Specify the interface properties and select one or more of the following checks and their defined "OK" status.

Parameters:

interface numeric

Network interface to be monitored.

*RST: 1

name
string
Name for the interface (optional).

errormessage
string
An additional error message that is shown if the status fails (optional).

returnstatus
WARNING | CRITICAL
Return value if the check fails (optional).

speed
numeric
Speed of the network interface, e.g. 100, 1000 Mbit/s (optional).
*RST: 1000
Default unit: MBit/s

op_status
UP | DOWN | TESTING | UNKNOWN | DORMANT | NOTPRESENT | LOWERLAYERDOWN
Check the operational status of the network interface (optional).

admin_status
UP | DOWN | TESTING
Administration status of the network interface (optional).

port_sec_mac
Checks if the MAC-based port security status of a device that is compatible with CISCO-PORT-SECURITY-MIB (optional).

port_sec_802
Checks if the 802.1-based port security status of a device that is compatible with CISCO-PAE-MIB (optional).

port_sec_ieee802
Checks if the PAE auth controlled port status of an interface is "AUTHORIZED" of a device that is compatible with the IEEE8021-PAE-MIB (optional).

Example:

```
- nw_interface:
    interface: 2
    speed: 1000
    op_status: UP
    admin_status: UP
    errormessage: "Failure on network interface for server"
    name: "server interface"
    returnstatus: "WARNING"
    port_sec_mac:

- nw_interface:
    interface: 3
    speed: 100
    port_sec_802:

- nw_interface:
    interface: 4
    port_sec_ieee802:
```

os_disk (Disk space)

Monitors available disk space.

Parameters:

include ['<drive or volume>' , 'drive or volume']

List of drives (on Windows) or volumes (on Linux) that are monitored (optional). If not set, R&S CHM monitors all disks or volumes.

*RST: none

thresholds warning | critical

Alert levels for available disk space (optional).
On Windows: **used** disk space.
On Linux: **free** disk space.
For more information about the `thresholds` syntax, see `thresholds` on page 98.

Range: 0 to 100
*RST: none (Windows) , 10 (Linux warning) , 20 (Linux critical)
Default unit: %

Example: For a Windows host

```
- os_disk:
    include: ['C', 'F']
    thresholds:
      warning: '80'
      critical: '90'
```

Example: For a Linux host

```
- os_disk:
    include: ['/', '/boot']
    thresholds:
      warning: '10:'
      critical: '5:'
```

os_memory (Memory usage)

Monitors RAM usage and detects when your operating system is about to swap.

Related parameters

- `thresholds`

Example:

```
- os_memory:
     thresholds:
       warning: '10:'
       critical: '5:'
```

os_process (Operating system process)

Monitors if a defined process is running on the system.

Parameters:

name

Name of the process. If at least one instance is found, the check is OK.

commandline

The check is performed against the command line of the process (optional).
If at least one instance is found, the check is OK.
On Linux: Regex is supported. For escaping special characters, use a backslash (\).
On Windows: Wildcards are supported (see: https://docs.microsoft.com/en-us/windows/win32/wmisdk/like-operator)

Example:

Checking for the process name:

```
- os_process:
    name: rsyslogd
```

Example:

Checking for the command line on Windows:

```
- os_process:
    name: svchost
    commandline: "%svchost%Unistack%"
```

Example:

Checking for the command line on Linux:

```
- os_process:
    name: icinga2
    commandline: icinga2.*daemon
```

os_service (Monitor Windows service status)

Monitors if a Windows service is in status "Running".

Parameters:

name

string

Specify the Windows "Service name". If the service is not in status "Running", the status is indicated as "CRITICAL" on the web GUI.

Example:

Monitor the status of the "Icinga2" service:

```
- os_service:
    name: "Icinga2"
```

passive (Aggregated host status)

Adopts the aggregated status from a logic function instance and shows this status on the web GUI.

Depending on the position in the configuration, passive has two meanings:

- If you specify `passive` as the first host check, it results in a logic host check.

- If you specify passive after other checks, it results in a service check with a logic function instance.

Parameters:

src_logic_id <log_func_inst>

Specify here one of the configured logic function instances. You can select from an instance that is configured in logic on page 48 or logic_id on page 93.

Example:

```
checks:
  - passive:
      src_logic_id: aggregation1
```

See also: Example in logic on page 48

ping (Host availability ("ping" check))

Checks the availability of a host. To do so, R&S CHM sends ICMPv4 or ICMPv6 requests to the hosts.

This check cannot verify if the R&S CHM service runs on an agent. To check this property, use chm_agent_connection, see chm_agent_connection on page 100.

Related parameters

- thresholds

Parameters:

threshold Thresholds for returned values of the ping command, i.e. rta and pl.

rta warning | critical

Round-trip average time (optional).

*RST: 3000 | 5000
Default unit: ms

pl warning | critical

Package loss (optional). Since five packages are sent, we recommend specifying one of the values 0, 20, 40, 60, 80, or 100.

*RST: 80 | 100
Default unit: %

Example:

```
checks:
  - ping:
      threshold:
        rta:
          warning: '500'
          critical: '1000'
        pl:
          warning: '60'
          critical: '80'
```

snmp (SNMP OID check)

Checks individual SNMP OIDs of a host for their return value. R&S CHM shows the status of the host with optional status message on the web GUI.

Status indication on the web GUI:

- "OK" if the returned value matches the expected value.
- "CRITICAL" if the returned value does not match the expected value.

Related parameters

- `snmp_connection`

Parameters:

oid
: string

  The SNMP OID to be checked.

expected
: string

  The expected return value.

okmessage
: string

  Show this message if the returned value matches the expected value.

criticalmessage
: string

  Show this message if the returned value does not match the expected value.

hwinfo
: true

  If you specify `hwinfo: true`, R&S CHM queries the SystemDescr OID and shows it on the web GUI > "Host" > "Result" (optional). The OID contains some basic information like the firmware version (if applicable). The check always returns as "OK".

Example:

```
checks:
  - snmp:
      snmp_connection:
        version: 2
        community: public
      oid: ".1.3.6.1.4.1.9.9.500.1.2.1.1.6"
      expected: "4"
      okmessage: "Cisco Switch State is READY"
      criticalmessage: "Cisco Switch State NOT READY"
```

The following is output on the web GUI if the check was successful:
"OK - Cisco Switch State is READY"

snmp_hostalive (Host availability ("snmp_hostalive" check))

Checks the availability of a host. To do so, the check sends an SNMP `GetNext` request targeting some OID close to the MIB root to the target host. If the host sends a response without SNMP error indication or status, the host is considered to be up and running.

You can use the check to determine if a host is "UP" or "DOWN" if ICMP is blocked in a system by a firewall.

Related parameters

- `snmp_connection`

Example:

```
checks:
  - snmp_hostalive:
      snmp_connection:
        port: 1234
        community: public
```

snmp_time (Check time offset to R&S CHM host)

Compares the time of a device with the time of the R&S CHM host using SNMP. The check supports all SNMP versions and can use all SNMP arguments.

Related parameters

- `snmp_connection`
- `thresholds`

Parameters:

tzoffset
numeric
Offset between the remote device and the R&S CHM host (in min).

localtime
boolean
Comparison method.

true
Compares remote time with the local time of the R&S CHM host.

false
Compares remote time with UTC.

thresholds
Thresholds for this status check.

offset
warning | critical
Thresholds for the time offset between device and server (in s).
*RST: 5 | 10
Default unit: s

Example:

```
checks:
  - snmp_time:
      tzoffset: 60
      localtime: false
      thresholds:
        offset:
          - warning: '20'
          - critical: '60'
```

spectracom_time (Spectracom time)

Monitors a Spectracom SecureSync time server via SNMP.

Checked values

- Status of AC and DC power supply
- Major and minor alarms
- GPS reference antenna status
- GPS reference time validity
- System synchronization status
- System holdover status
- Number of satellites

Supported MIBs

- SPECTRACOM-SECURESYNC-MIB

Tested devices

- Spectracom SecureSync GT4030

Related parameters

- `snmp_connection`

Parameters:

name
string
The name for the device that is shown in the check results.
*RST: Spectracom SecureSync

no_acpower
Do not check the AC power status (optional).

no_dcpower
true
Do not check the DC power status (optional). This key requires that you specify `no_dcpower: true` in the configuration file.

no_minor_alarm
Do not check for minor system alarms (optional).

no_major_alarm
Do not check for major system alarms (optional).

no_ref_time_validity
Do not check the GPS ref time validity (optional).

no_sync_state
Do not check the system sync status.

no_holdover_state
Do not check the system holdover status (optional).

no_ref_antenna_state Do not check the GPS ref antenna status (optional).

no_tracked_satellites Do not check the number of tracked satellites (optional).

thresholds Check-specific alert levels (optional). For more information about the threshold syntax, see `thresholds` on page 98.

tracked_satellites warning | critical

Defines the `thresholds` for the number of tracked satellites (optional).

boolean

*RST: warning: '5:' , critical: '3:'

Example:

```
- spectracom_time:
    name: Spectracom GT4030
    no_dcpower:
    thresholds:
      tracked_satellites:
        - warning: '5:'
        - critical: '3:'
```

ssh (Establish SSH connection)

This check attempts to establish an SSH connection to the specified host and port.

The check results:

- If the connection is successful:
  - The check returns "OK".
  - The check returns "CRITICAL" in combination with `ssh_negate: True`.
- If the connection fails:
  - The check returns "CRITICAL"
  - The check returns "OK" in combination with `ssh_negate: True`.

Parameters:

ssh_port integer

The port number on which the SSH server is listening (optional). Default port is 22.

ssh_timeout integer

The timeout (in s) for the SSH connection (optional). Default is 1 s.

ssh_negate True | False

If set to `True`, the check returns "OK" if the SSH server is unreachable or the connection is refused. Default is `False`.

Example:

```
checks:
  - ssh:
      ssh_port: 22
      ssh_timeout: 10
      ssh_negate: False
```

Example:

```
checks:
  - ssh:
      ssh_timeout: 5
```

Example:

```
checks:
   - ssh:
       ssh_negate: True
```

synology (Synology NAS)

Monitors various aspects of a Synology NAS via SNMP.

Checked values

- Global temperature
- Power unit status
- Fan status
- Status of all drives
- RAID status

Related parameters

- `snmp_connection`

Parameters:

no_temperature
: true
Do not check overall thermal environment condition (optional).

no_power_status
: true
Do not check global power status (optional).

no_fan_status
: true
Do not check the fan status (optional).

no_disks
: true
Do not check the disk status (optional).

no_raid
: true
Do not check the RAID status (optional).

Example:

```
- synology:
    no_fan_status: true
```

system_state (Enable client interface and check logic)

Enables the Windows client interface and the check logic. The check shows the aggregated state of the entire system on the web GUI. The clients connect to this check and replicate the status to the notification icon. If there is a change in any check, the `system_state` check mirrors that state.

Example:

```
hosts:
  - name: host1.de
    tags: [chm]
    checks:
      - ping:
      - system_state:
```

How to: Section 4.3, "Installing R&S CHM clients", on page 25

---

tcp (TCP port connectivity check)

Checks if a TCP port is open and reachable from the R&S CHM host.

Parameters:

port_number The port to be checked.

Example:

```
- tcp:
    tcp_port: 22
```

---

tmr_radio (TMR-MIB compatible radio)

Shows the mode of TMR-MIB compatible radios. Such devices can be in normal mode or control mode. The check returns the state of the radio by using the plugin output. For *normal*, the web GUI shows "OK - Normal Mode", for *control*, it shows "OK - Control Mode". If anything else is returned, the web GUI shows "UNKNOWN - Unknown Mode (n)".

Example:

```
checks:
  - tmr_radio:
```

---

trustedfilter (Monitor R&S TF5900M trusted filter IP)

Monitors the status of the R&S TF5900M trusted filter IP firewall, which secures the boundaries of networks with different classified material domains:

- Status power supply unit 1/2
- Status power fan unit 1/2
- Status internal voltage
- Status internal temperature
- Error status
- Activity state
- Status log fill level

Related parameters

- `snmp_connection`

Example:

```
checks:
  - trustedfilter:
      snmp_connection:
        version: 2
        community: public
```

ups (Uninterruptible power supply - RFC1628-compatible)

Monitors a UPS that is compatible to RFC1628 via SNMP.

Related parameters

- `snmp_connection`
- `thresholds`

Select one of the following checks. Each check returns a single metric.

Parameters:

alarms

Checks the present number of active alarm conditions.
In combination with `thresholds`, R&S CHM generates an alert.

secondsonbattery

Checks if the unit is running on battery power? If not, the UPS returns zero.
If the unit is not running on battery power the following is checked, whichever is less:
The elapsed time since the UPS last switched to battery power.
– or –
The time since the network management subsystem was last restarted.
In combination with `thresholds`, R&S CHM generates an alert.

Default unit: s

minutesremaining

Checks estimated time to battery charge depletion under the present load states in the following cases:
The utility power is off and remains off.
– or –
The utility power is going to be lost and remains off.
In combination with `thresholds`, R&S CHM generates an alert.

Default unit: min

thresholds

Check-specific alert levels. For more information about the threshold syntax, see `thresholds` on page 98.

Example:

```
- ups:
   alarms:
   thresholds:
      warning: '0:'
      critical: '0:'
- ups:
   secondsonbattery:
   thresholds:
      warning: '0:'
      critical: '0:'
- ups:
   minutesremaining:
   thresholds:
      warning: '~:20'
      critical: '~:10'
```

vmware (VMware ESXi/vcenter server inventory)

Monitors a VMware ESXi/vcenter server, e.g. datastores. You can specify up to four checks for a host.

Available checks

- Alarms
- Datastore usage
- CPU usage
- Memory usage

Related parameters

- `thresholds`

Parameters:

user
string
The user name that is used to log in at the server.

insecure
false | true
Checks the server certificate (optional). The server certificate is checked (`'false'`) or *not* checked (`'true'`).
*RST: false

type
alarm | datastore | hostsystem
The entity type of the monitored object.

alarm
Currently not acknowledged alarms on the alarm list result in an alert with the severest alarm state, i.e. warning or critical.

datastore
Gets used disk space on datastore objects.

hostsystem
Gets CPU and memory usage on all HostSystem objects, i.e. ESX(i) hosts. See also the thresholds parameter.

id

string

The unique identifier for the monitored object (optional). If no `id` is given, all objects of the specified type are checked. E.g., for datastores, `id` is the name of the datastore. The parameter is not supported for `alarm` and `hostsystem`.

port

numeric

Port of the VMware vSphere API (optional).

*RST: 443

thresholds

cpu | memory

Check-specific alert levels (optional). For more information about the `thresholds` syntax, see `thresholds` on page 98. The thresholds for the datastore usage define the used datastore space (in %).

cpu
Usage of CPU (in %).

memory
Usage of RAM (in %).

Example:

```
- vmware:
    user: axolotl
    type: datastore
    id: mydatastore
    thresholds:
      warning: '90'
      critical: '95'
- vmware:
    user: axolotl
    type: alarm
- vmware:
    user: axolotl
    type: hostsystem
    thresholds:
      cpu:
        warning: '90'
        critical: '95'
      memory:
        warning: '98'
        critical: '99'
```

windowsupdateage (Windows security update)

Checks if at least one Windows security update was installed within the last given number of days.

Related parameters

- `thresholds`

Parameters:

thresholds

warning | critical

Alert levels for the age of the definition files (optional).

*RST: critical: '20'
Default unit: d

Example:

```
- windowsupdateage:
    thresholds:
      critical: '100'
```

8 YAML configuration examples

This chapter provides some examples for configuration of hosts and services in the YAML configuration file.

8.1 R&S CHM host configuration

The following YAML code snippet shows the top part of the configuration file with the definition of the R&S CHM host. For configuration details, see Section 6.4, "Configuring hosts", on page 41.

```
hosts:
  - name: host1.de
    tags: [chm]
    authentication:
      monitoring:
        - ldap:
            server: ldapserv.ourlocal.net
            port: 35636
            encryption: ldaps
            base_dn: ou=ldap_users,dc=ldapserv,dc=ourlocal,dc=net
            user_class: user
            user_name_attr: sAMAccountName
            bind_dn: service_user
            bind_pwd_path: ldap/service_user
    authorization:
      monitoring:
        roles:
          admin:
            permissions:
              - check
              - acknowledge
              - comment
              - downtime
            users:
              - admin
              - armin
            groups:
              - G_Admins
              - G_Armins
          superoperator:
            permissions:
              - acknowledge
            users:
              - supop
          special:
    connections: [local]
    hostgroups: [monitoring, control]
    checks:
      - icinga2_cluster:
          checkgroups: [cluster, buster]
      - dhcp:
          displayname: Check our awesome DHCP servers
          servers: 192.168.1.253, 192.168.1.254
          interface: eth0
      - dns:
          displayname: Check our insane DNS servers
          lookup: somehosttolookup.ourlocal.net
          server: 192.168.1.254
          answers: 192.168.1.10, 192.168.1.11
          authoritative: true
          accept_cname: true
          timeout: 15
          thresholds:
            warning: '5'
            critical: '10'
```

8.2 Linux host configurations

Here, you can find some examples for Linux host configurations.

Example: host3.de

```
- name: host3.de
    connections: [icinga2_linux]
    checks:
      - os_process:
          name: test
      - load:
          thresholds:
            load1:
              warning: '9'
              critical: '10'
            load5:
              warning: '8'
              critical: '9'
      - os_disk:
          include: ['/', '/boot']
          thresholds:
            warning: '10:'
            critical: '5:'
      - ntp_time:
          server: ntpserver.example.com
          thresholds:
            warning: '1'
            critical: '2'
```

Example: chm2-test-linux-node.rsint.net

```
- name: chm2-test-linux-node.rsint.net
    connections: [icinga2_linux]
    hostgroups: [oumuamua]
    checks:
      - ping:
      - os_memory:
      - os_disk:
          include: ['/', '/boot']
          thresholds:
            warning: '10:'
            critical: '5:'
      - nport:
          checkgroups: [water, earth, fire, air]
          snmp_connection:
            version: 3
            context: nport
            secname: rsadmin # lookup of passwords in password store
            authproto: MD5
            privproto: DES
            port: 1234
          serial_port: 1
          cts: LOW
          errormessage: "GENERATOR FAILED"
          name: "GENERATOR INPUT"
          returnstatus: "WARNING"
```

9 Troubleshooting

This section informs about problems that can occur and provides basic troubleshooting procedures. Problems that apply to the web GUI probably cannot be resolved by operators or administrators due to missing privileges. Then, contact the system administrator to resolve these problems.

9.1 Web GUI is unavailable

If the web GUI is unavailable, possibly the services are not up and running on the R&S CHM system status monitoring host.

Resolution

► Check if the services are running on the R&S CHM host:

Access authorization: *root*

- `# sudo systemctl status chm`
- `# sudo systemctl status icinga2`

Access authorization: *root*

► Restart these services on the R&S CHM host:

```
# sudo systemctl restart chm
# sudo systemctl restart icinga2
```

See also: "To edit the configuration file" on page 40.

9.2 Web GUI shows message Wrong SNMP PDU digest

Or you can see the SNMP error "No SNMP response received before timeout".

Resolution

► Check the SNMP settings on the device, i.e. `snmp_connection` keys `context`, `authpass`, `privpass`, `authproto`, etc. The configuration in the `chm.yaml` file does not match the monitored device.

See also: `snmp_connection` on page 95

9.3 Web GUI shows 404 error

This error is a standard HTTP error message code. It means that the website that you were trying to reach could not be found on the server. One of the possible causes is that the LDAP server is not reachable.

Resolution

1. Ensure that the LDAP server is up and running.
2. If you cannot fix the problem, consider disabling LDAP in the YAML configuration to access the web GUI using a local user account.
   To disable LDAP, see Section 6.5, "Configuring web GUI users", on page 57.

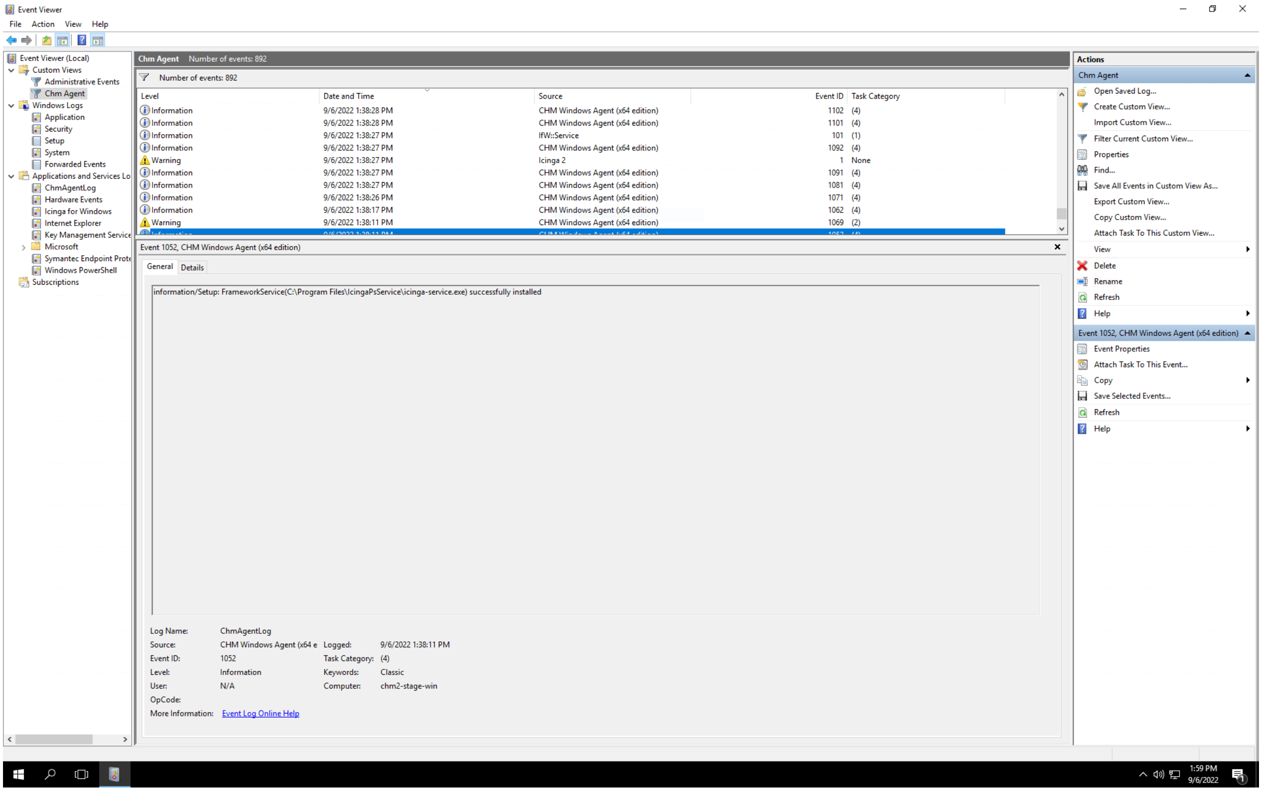
9.4 Troubleshooting installation problems on Windows agents

If you experience problems during installation of Windows agents using the CHM_Windows_Agent_<version>.exe installer, you can find troubleshooting information in the Windows Event Viewer.

1. Select ⊞.
2. Type *event viewer*.

   The Event Viewer opens.
3. In the left navigation area, select "Custom Views" > "CHM Agent".

   The events from the R&S CHM Windows agent installation are listed.



Chm Agent Number of events: 892

Number of events: 892

Level	Date and Time	Source	Event ID	Task Category
Information	9/6/2022 1:38:28 PM	CHM Windows Agent (x64 edition)	1102	(4)
Information	9/6/2022 1:38:28 PM	CHM Windows Agent (x64 edition)	1101	(4)
Information	9/6/2022 1:38:27 PM	IfW::Service	101	(1)
Information	9/6/2022 1:38:27 PM	CHM Windows Agent (x64 edition)	1092	(4)
Warning	9/6/2022 1:38:27 PM	Icinga 2	1	None
Information	9/6/2022 1:38:27 PM	CHM Windows Agent (x64 edition)	1091	(4)
Information	9/6/2022 1:38:27 PM	CHM Windows Agent (x64 edition)	1081	(4)
Information	9/6/2022 1:38:26 PM	CHM Windows Agent (x64 edition)	1071	(4)
Information	9/6/2022 1:38:17 PM	CHM Windows Agent (x64 edition)	1062	(4)
Warning	9/6/2022 1:38:11 PM	CHM Windows Agent (x64 edition)	1069	(2)

Event 1052, CHM Windows Agent (x64 edition)

```
information/Setup: FrameworkService(C:\Program Files\IcingaPsService\icinga-service.exe) successfully installed
```

Log Name:	ChmAgentLog		
Source:	CHM Windows Agent (x64 e	Logged:	9/6/2022 1:38:11 PM
Event ID:	1052	Task Category:	(4)
Level:	Information	Keywords:	Classic
User:	N/A	Computer:	chm2-stage-win
OpCode:			
More Information:	Event Log Online Help		

Figure 9-1: Event Viewer - logs from R&S CHM (example)

9.5 Contacting customer support

Technical support – where and when you need it

For quick, expert help with any Rohde & Schwarz product, contact our customer support center. A team of highly qualified engineers provides support and works with you to find a solution to your query on any aspect of the operation, programming or applications of Rohde & Schwarz products.

Contact information

Contact our customer support center at www.rohde-schwarz.com/support, or follow this QR code:

Figure 9-2: QR code to the Rohde & Schwarz support page

Glossary: Abbreviations and terms

A

AES: Advanced encryption standard

agent: An R&S CHM agent instance on Windows or Linux hosts that sends its monitoring results to an R&S CHM host. Read complete definition: Section 6.10, "Configuring distributed monitoring", on page 78

API: Application programming interface

B

bind user: The user that is necessary to access the user and group information at the LDAP server.

C

CA: Certificate authority

CentOS: Linux distribution that is derived from Red Hat Enterprise Linux (RHEL). Linux is necessary for running the R&S CHM host software.

client: A client is a host that runs the R&S CHM client application. It is intended for running the web GUI with additional features.
See Section 4.3, "Installing R&S CHM clients", on page 25.

CPU: Central processing unit

CSR: Certificate signing request

CSS: Cascading style sheets

CTS: Clear to send

D

DES: Data encryption standard

DISA: Defense Information Systems Agency

DKN: Digitales Kommunikationsnetz (digital communications network)

DN: Distinguished name

DNS: Domain network service

DSR: Data set ready. A DSR signal change indicates that the power of the data communication equipment is off.

DTR: Data terminal ready

F

FIPS: Federal information processing standard. FIPS standards establish requirements, e.g. for ensuring computer security and interoperability.

FQDN: Fully qualified domain name

G

gb2pp: A Rohde & Schwarz proprietary network protocol

GPG: GNU privacy guard

gRPC: General-purpose remote procedure calls.

GUI: Graphical user interface

H

HA: High availability

HDD: Hard disk drive

HMAC: Hash-based message authentication code

host: A host is an independent device in the system, which is addressed and monitored by R&S CHM. A host is, e.g. a Windows PC or a Linux virtual machine, or a device that you monitor using SNMP.

HP iLO: Integrated Lights-Out interface from Hewlett-Packard for configuration, update and remote server operation

HTML5: Hypertext mark-up language, version 5

HTTP: Hypertext transfer protocol

HTTPS: Hypertext transfer protocol secure

HUMS: Rohde & Schwarz health and utilization monitoring system

I

ICMP: Internet control message protocol

iDRAC: Integrated Dell remote access controller

ISO image: A disc image that contains everything that would be written to an optical disc. The ISO image contains the binary image of the optical media file system.

J

JSON: JavaScript Object Notation

K

KDC: Key distribution center, it handles authentication, ticket granting and holds a database with all the principals.

Kerberos: A computer network authentication protocol.

Kerberos ticket: A certificate that is issued by an authentication server and encrypted using the server key. There are two types of tickets, TGT and ST.

keytab: Short for "key table". A file that stores long-term keys for one or more principals. Can be extracted from principal database on KDC server.

L

LAN: Local area network

LCD: Liquid crystal display

LCSM: Lifecycle software manager

LDAP: Lightweight directory access protocol

LXI: LAN extensions for instrumentation

M

MAC: Media access control

master: R&S CHM host instance that is located in the top-level subsystem. Read complete definition: Section 6.10, "Configuring distributed monitoring", on page 78

MD5: Message digest algorithm 5

MIB: Management information base. Collection of objects in a virtual database that allows network managers using Cisco IOS software to manage devices such as routers and switches in a network.

N

NAS: Network attached storage

NAVICS: Navy integrated communication system

NSS: Name service switch. Provides a central configuration store where services can look up sources for various configuration and name resolution mechanisms.

NTP: Network time protocol

O

OID: Object identifier. An address that uniquely identifies managed devices and their statuses. The SNMP protocol uses OIDs to identify resources that can be queried, among other things.

P

package cache: The package cache folder is a system folder. By default, it is located on the drive where your operating system is installed. The folder is used by applications to store settings, caches, installers and packages.

PAE: Port access entity

PDF: Portable document format. Frequently used file format for saving and exchanging documents.

PDU: Power distribution unit

PEM: Privacy-enhanced mail; a container format that can include only a public certificate or an entire certificate chain, including public key, private key, and root certificates.

PKI: Public key infrastructure

principal: A kerberos principal is a unique identity to which kerberos can assign tickets.

R

RAID: Redundant array of independent disks.

RAM: Random access memory

S

satellite: R&S CHM host instance that is not placed in the top-level subsystem. Read complete definition: Section 6.10, "Configuring distributed monitoring", on page 78

SHA: Secure hash algorithm

SIP: Session initiation protocol

SNMP: Simple network management protocol. It allows devices to exchange monitoring and managing information between network devices.

SSD: Solid state drive

SSH: Secure shell

SSO: Single sign-on

SSSD: The system security services daemon is a system daemon.

ST: Service ticket. Obtained from the TGS.

subsystem: A subsystem is at least one R&S CHM node that is grouped with any number of non-R&S CHM hosts or devices. Each R&S CHM host instance in a subsystem provides its own web GUI.

T

TCP: Transmission control protocol

TGS: Ticket granting server. A logical KDC component that is used by the Kerberos protocol as a trusted third party.

TGT: Ticket granting ticket. A user authentication token issued by the KDC that is used to request access tokens from the TGS for specific resources or systems that are joined to the domain.

TLS: Transport layer security

U

UPS: Uninterruptible power supply

UTC: Universal time coordinated

V

VM: Virtual machine

W

web GUI: Short for R&S CHM web GUI. The web GUI runs in a browser. It shows all information collected by R&S CHM. If you run the web GUI on a Windows client, you can take advantage of additional features.
See Section 4.3, "Installing R&S CHM clients", on page 25.

X

XML: Extensible markup language

Y

YAML: YAML™ ain't markup language

Glossary: Specifications

R

RFC 5424: The Syslog Protocol

RFC1213: Management Information Base for Network Management of TCP/IP-based internets: MIB-II

RFC1628: UPS Management Information Base

List of keys

Index