Güralp Data Box (GDB)

Güralp Data Centre (GDC) in a Box

Technical Manual

Document Number: MAN-GDC-0001

Designed and manufactured by
Güralp Systems Limited
3 Midas House, Calleva Park
Aldermaston, RG7 8EA

1. Preliminary Notes

1.1. Proprietary Notice

The information in this document is proprietary to Güralp Systems Limited and may be copied or distributed for educational and academic purposes but may not be used commercially without permission.

Whilst every effort is made to ensure the accuracy, completeness, and usefulness of the information in the document, neither Güralp Systems Limited nor any employee assumes responsibility or is liable for any incidental or consequential damages resulting from the use of this document.

1.2. Notes Cautions and Warnings

Notes, Cautions and Warnings are displayed and defined as follows:

Note: Indicates a procedural or advisory information

Caution: Highlights a condition or action that, if not followed, may result in equipment damage, reduced performance, or loss of data.

Warning: Alerts to a hazard that could result in personal injury or serious damage to the equipment if ignored.

1.3. Manuals and Software

All manuals and software referenced in this document are, unless stated otherwise, available from the Güralp Systems Ltd website: www.guralp.com.

1.4. Conventions

Throughout this manual, examples of command-line interactions are shown in a fixed-width typeface:

```
fixed-width example
```

Commands you must type appear in **bold**:

```
bold fixed-width example
```

Variable elements you must replace with values specific to your configuration are blue and shown in bold:

```
Bold blue fixed-width example
```

Putting these together into a single example:

```
System prompt: User input variable parameter
```

On-screen buttons referenced in procedures are shown using the following style:

Example Button

2. Overview

The Güralp Data Box (GDB) serves as an all-in-one device which comes preconfigured with Güralp Systems Data Centre (GDC) acquisition software package, Güralp Discovery software, and all necessary dependencies. Güralp Data Centre software package (acquisition software package) consists of several applications providing system state of health monitoring, data collection and distribution, and remote configuration capabilities. GDC is scalable as a standalone cloud service.

For the purpose of this document, GDC will imply reference to the software package, and instructions are applicable to both GDB and the standalone GDC. This document describes how to install, operate and configure a Güralp Data Box, or selected components of the software package.

2.1. Software Components

List of all applications present in the software components diagram:

- IRIS ringserver
- slinktool
- slink2dali
- Güralp Data Centre controller service
- Güralp Data Centre monitor service
- Güralp responder service
- Güralp Discovery application

Optional components:

- Earthworm client
- SeiscomP3 client

3. Installing GDC Acquisition Software Package

3.1. Introduction

This section provides instructions of how to install Güralp Data Centre acquisition software package with required dependencies. Software package is provided in a form of a set of RPM files that in this document are installed using dnf package manager on Red Hat 8.

Note: All these packages are pre-installed on the Guralp Data Box (GDB). Follow the instructions below only if you are installing Guralp Data Centre (GDC) on your own machine.

3.2. Operating System Requirements

Güralp Systems Data Centre software package has been tested on the following x86_64 platforms:

- Red Hat Enterprise Linux 8 (or equivalent, e.g. Rocky Linux 8 or AlmaLinux 8)
- Amazon Linux 2

Support for other platforms might be considered upon request (minimum system dependency requirements are: systemd v239, Qt v5.12.5, polkit v0.115).

3.3. Software Package Content

Software package provided contains 4 RPM files:

- **guralp-datc-0.5-1.el8.x86_64.rpm**

  Installs the main components of the Data Centre: IRIS ringserver, Güralp responder, nginx reverse proxy and Python Flask server to handle HTTP requests from Discovery and the Data Centre controller and monitor binaries.

- **libmseed-2.19.6-0.1.el8.x86_64.rpm**

  Installs libmseed library providing MiniSEED support for SEED related parts of the software solution.

- **slink2dali-0.7b-0.1.el8.x86_64.rpm**

  Installs slink2dali executable required to convert SeedLink data received from the seismic station to DataLink data consumed by the IRIS ringserver.

- **slinktool-4.3b-0.1.el8.x86_64.rpm**

  Installs slinktool executable required to measure the data latency.

3.4. Installation

Software package is provided in a set of RPM files that should be installed using the operating system package manager. This document describes installation procedure on Red Hat Enterprise Linux 8 with dnf package manager.

Installation requires root privileges and access to the RedHat packages repository.

3.4.1. Install libmseed

Install libmseed package from the provided RPM using dnf package manager by executing the following command:

```
sudo dnf install libmseed-2.19.6-0.1.el8.x86_64.rpm
```

3.4.2. Install slinktool

Install slinktool package from the provided RPM using dnf package manager by executing the following command:

```
sudo dnf install slinktool-4.3b-0.1.el8.x86_64.rpm
```

3.4.3. Install slink2dali

Install slink2dali package from the provided RPM using dnf package manager by executing the following command:

```
sudo dnf install slink2dali-0.7b-0.1.el8.x86_64.rpm
```

3.4.4. Install Güralp Data Centre Software

Install Güralp Data Centre software package from the provided RPM using dnf package manager by executing the following command:

```
sudo dnf install guralp-datc-0.5-1.el8.x86_64.rpm
```

3.4.5. Verification

Each installation step should complete without failures and all of the required dependencies should be pulled from the package repository. Please contact Güralp support in case of any problems.

Successful installation should result in all of the key services to be enabled and running in the operating system what can be verified by executing the following commands:

- **For Güralp responder:**

  `systemctl status guralp-responder.service`

  Reported status should indicate the service is **active** and **running**.

- **For Güralp Data Centre monitor:**

  `systemctl status guralp-monitor.service`

  Reported status should indicate the service is **active** and **running**.

- **For Güralp Data Centre controller:**

  `systemctl status guralp-controller.service`

  Reported status should indicate the service is **active** and **running**.

- **For IRIS ringserver:**

  `systemctl status iris-ringserver.service`

  Reported status should indicate the service is **active** and **running**.

- **For Python Flask Server:**

  `systemctl status python-server.service`

  Reported status should indicate the service is **active** and **running**.

- **For Nginx reverse proxy:**

  `systemctl status nginx`

  Reported status should indicate the service is **active** and **running**.

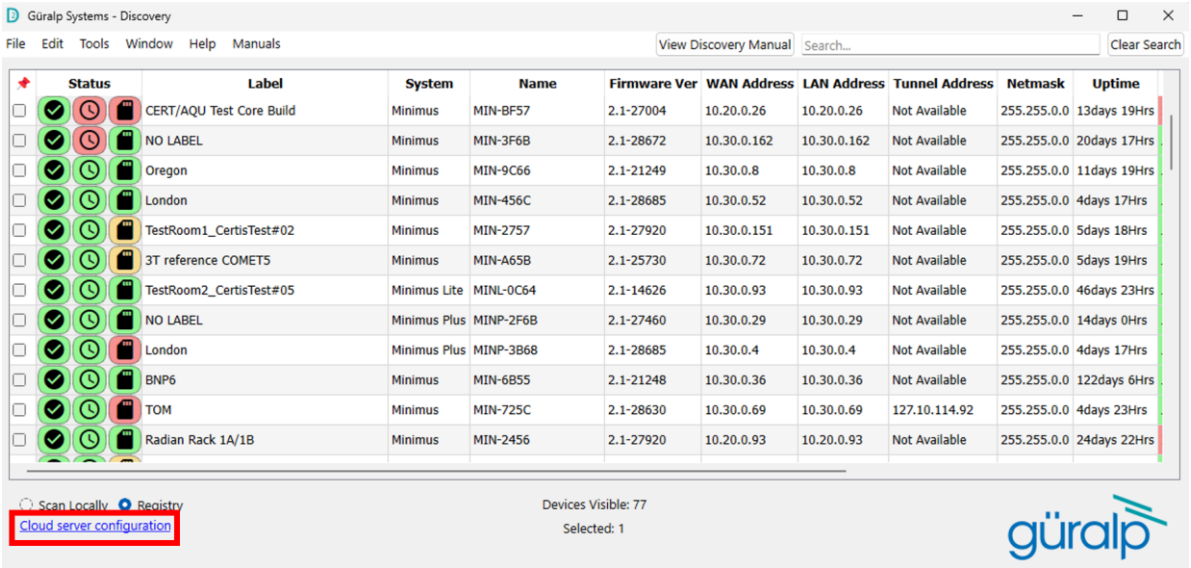
With these services active and running you should be able to view and interact with your GDC instance via Güralp's Discovery desktop application.

3.5. Güralp Discovery Application

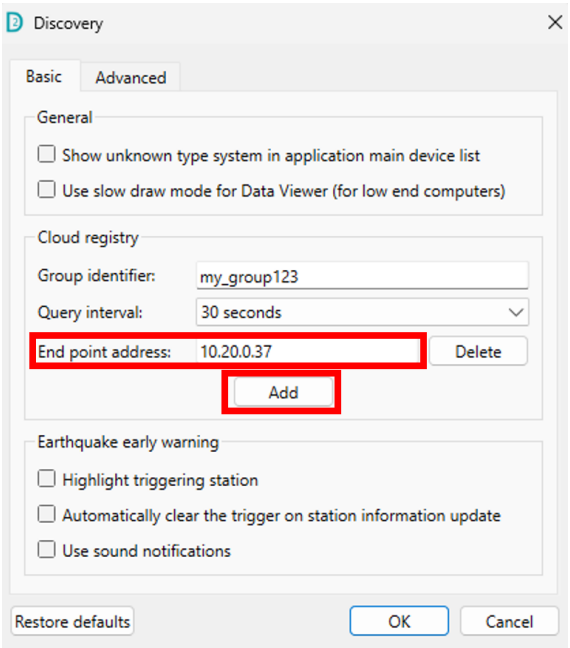
Güralp Discovery is a standalone application dedicated to run in a desktop environment with Windows, Linux or Mac operating system. Intended as a primary interface between the user and GDC, the application provides multiple functionalities for controlling, diagnosing and monitoring Güralp Systems devices and software products. For the Data Centre acquisition software solution, Discovery can be used to achieve the following:

- Stream live data
- Download data
- Configure stations
- Monitor data centre state of health

First, Discovery must be configured to use the Data Centre IP address as a Cloud registry server. Discovery supports multiple cloud registry endpoints and these are configured under File/Settings menu action, also accessible by clicking on the “Cloud server configuration” label of the application main window.



To add a cloud registry endpoint, please click on an “Add” button in the “Cloud registry” section and provide endpoint hostname or IP address. On a GDB, the endpoint address is that of the GDB itself. Please note that hostnames will get automatically translated to IP address and stored as IPv4 address.

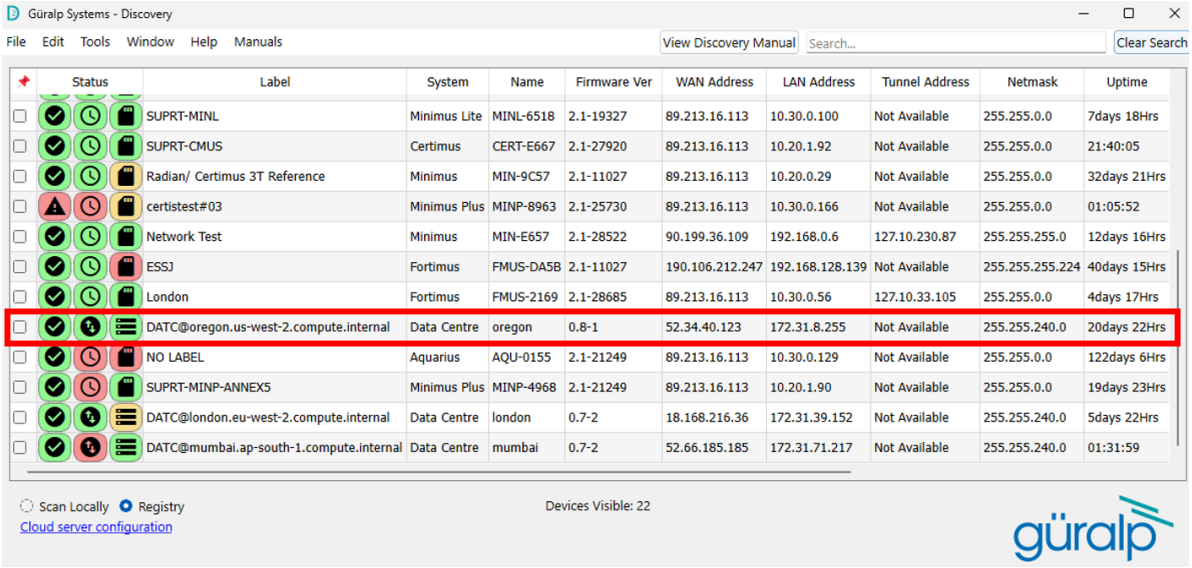


When all of the Data Centre instances are added to the list, click on the “Apply” button to confirm the changes.

It is important to set the “Cloud registry group identifier” field to reflect the configuration of the Data Centre Monitor and the Güralp seismic stations (e.g. Minimus or Fortimus). Misconfiguring this setting will cause the list of active devices to be blank in the “Registry” mode of the Discovery application.

To reduce the amount of network traffic, the frequency of state of health information packet requests from the Cloud server can be configured by changing the “Cloud query interval” setting.

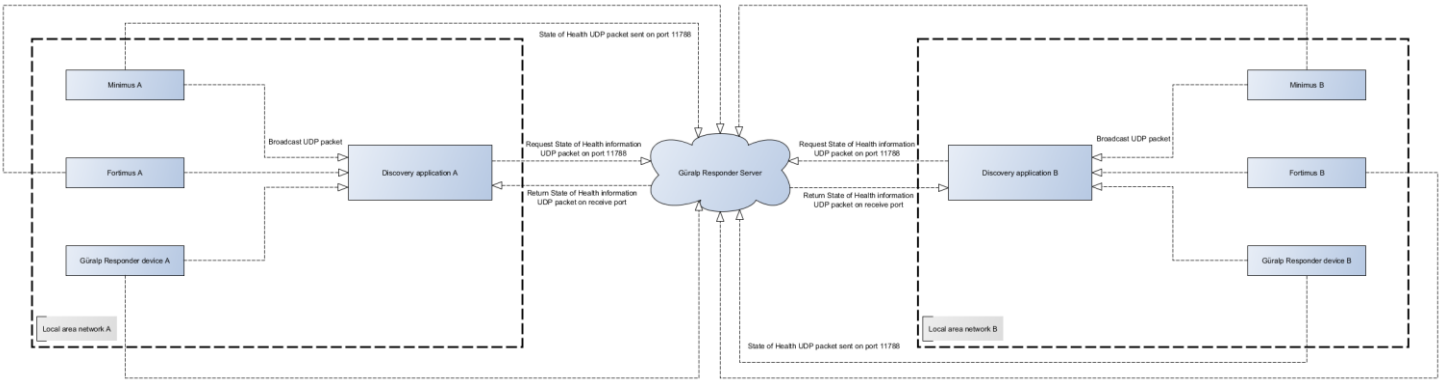
You should now be able to see your GDC instance or instances in Discovery:



The following section ‘Operation’, will walk through how to operate your GDC instances using Discovery.

4. Operation

4.1.1. State of Health (SOH)

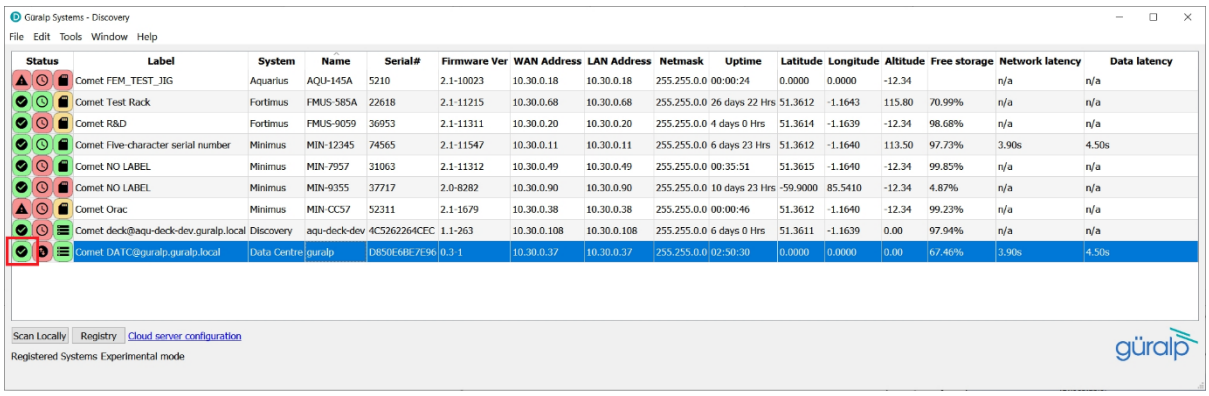


The graph above shows the state of health packet circulation in an environment with 2 local area networks and 1 Güralp responder server instance. The Data Centre software package includes Güralp responder service and the Data Centre acts as a server.

State of health information can be delivered to the Discovery application in 2 ways:

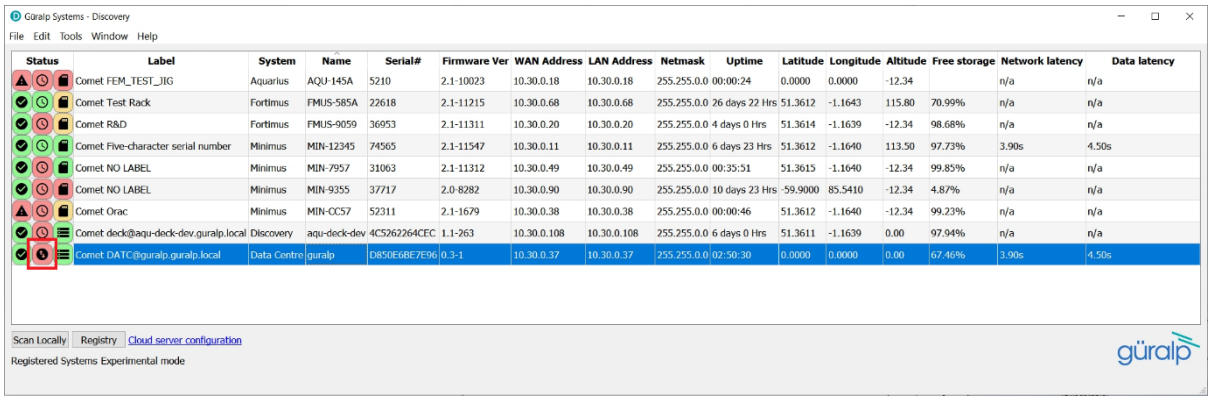
- In a local network, state of health information can be broadcasted by UDP packet sent on port 11788, Discovery is listening to the valid broadcast packets and lists the device in the applications main window table under “local” mode.
- State of health information can be sent directly to Güralp responder server where it is collected and distributed to the Discovery upon request. In “Registry” mode the application is querying the responder periodically for the latest information.

The Data Centre state of health can be monitored in the real time either in the Discovery desktop application main window by checking the status indication icons, and/or by accessing the dedicated Data Centre state of health widget.



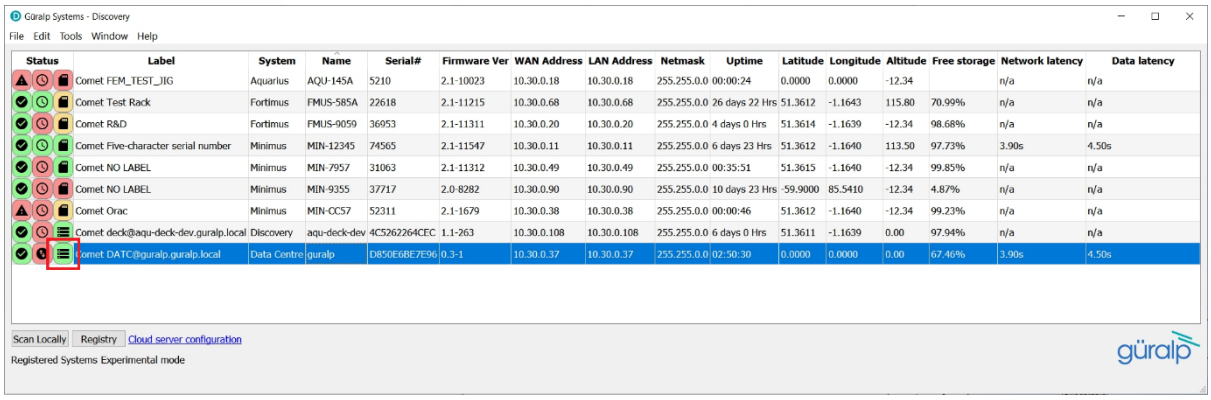
First status icon indicates the active state of the Data Centre. Active state traffic light colour scheme indicates the following:

- Green background – state of health information was received in last 30 seconds.
- Amber background – system is booting.
- Red background – state of health information was not received for more than 90 seconds.



Second status icon indicates the latency status of the data coming in to the Data Centre. Data Centre monitors the latency of all channels received and satisfied by the latency channels monitoring filter and sends the highest latency in the state of health information packet. The latency status traffic light colour scheme indicates the following:

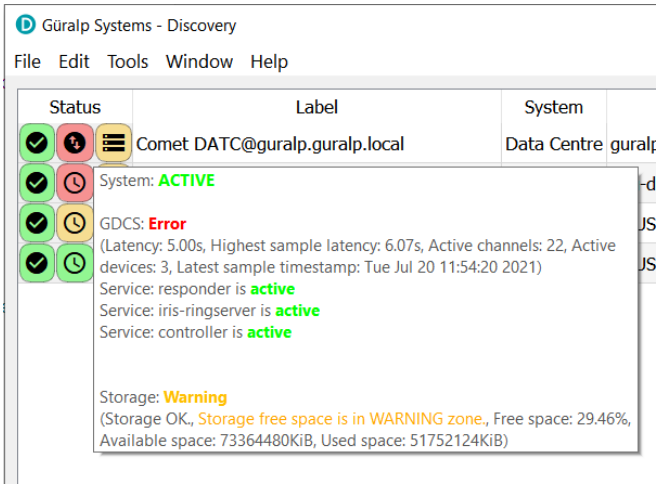
- Green background – the highest latency value is below 1 second.
- Amber background – the highest latency value is between 1 second and 1.5 second.
- Red background – the highest latency value is above 1.5 second.



The last, third status icon shows the Data Centre storage status. The traffic light colour scheme for the Data Centre storage indicates the following:

- Green background - more than 50% of disk space is free.
- Amber background – between 20% and 50% of disk space is free.
- Red background – less than 20% of disk space is free.

More descriptive tooltip is provided when hovering over the status icons of a given device:



4.1.2. Options for SOH Update Messages

A GDC instance can be configured to send updates regarding device State of Health (SoH) via email or SMS messages to a list of recipients. This can be done through editing the following config files on the GDB, or a computer running the Data Centre:

- `/opt/guralp/system_notification_list`
- `/opt/guralp/scripts/soh_mail`
- `/opt/guralp/scripts/soh_text`

The file `system_notification_list` contains a list of the instruments the user wishes to receive updates for, a list of flags for different SoH messages, and a list of email recipients (optional). The files `soh_mail` and `soh_text` contain configuration for SMTP and SMS service provider configuration respectively. To add a device to the `system_notification_list`, open the file in any text editor and add a new line for each device with the following syntax:

```
[device name];[bool];[bool];[bool];[bool]; [email]
```

For example:

```
MIN-1234;true;true;true;true; example@domain.com
```

Each of the true/false flags corresponds to a SoH message the registry may be configured to send for that device. They are as follows:

- Device has not made contact with registry server in over 120 seconds
- Device has reestablished contact with registry server after 1800s without contact
- Device has been added to registry server
- Any change to device status

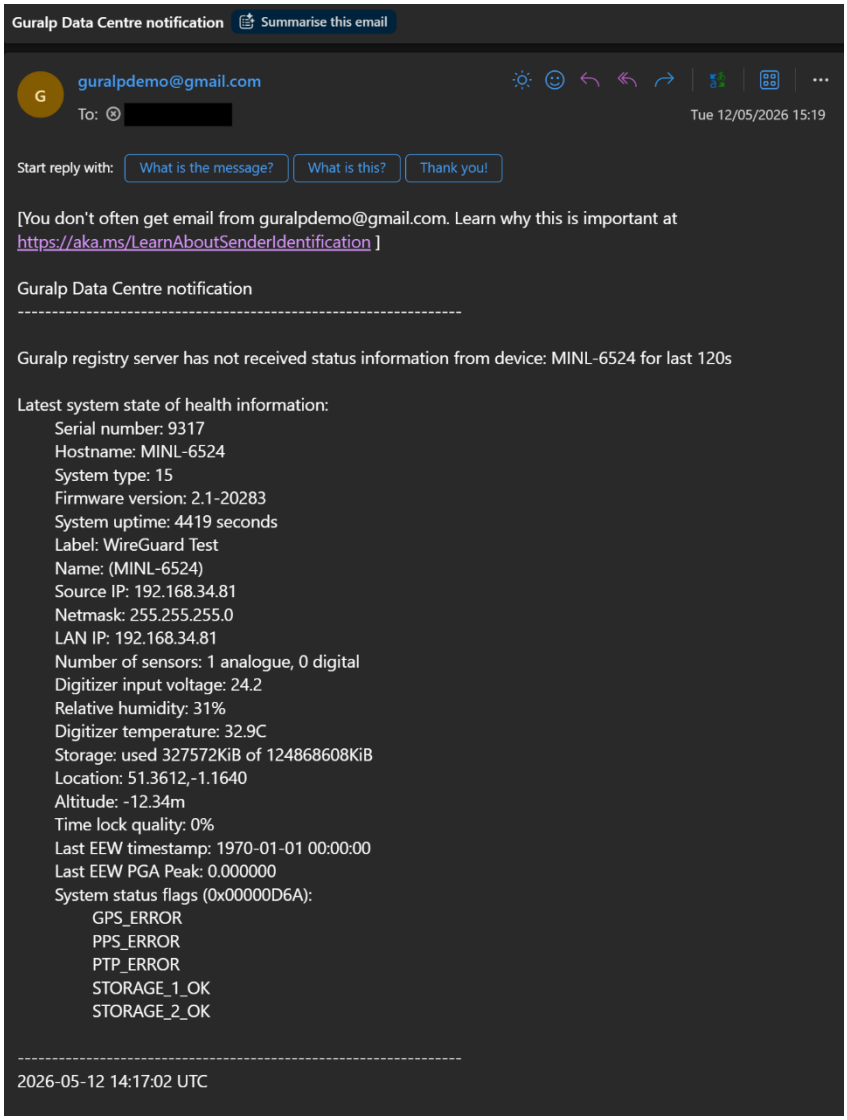
The file `soh_mail` requires the user to configure a SMTP server (e.g.: smtp.gmail.com) and uses a CURL command to request an email. The `soh_mail` file can be configured with recipients as follows:

```
curl –url <smtp-url> --mail-from <sender> --mail-rcpt <recipient> --user
<username:password> --upload-file <file>
```

The GDB does not host its own SMTP server and must be configured with one of the users choosing. An example configuration is provided using Google Gmail SMTP, with the entire command on one line:

```
curl --url ‘smtps://smtp.gmail.com:465’ \
        --mail-from ‘guralpdemo@gmail.com’ \
        --mail-rcpt ‘example@guralp.com’ \
        --user ‘guralpdemo@gmail.com:**********’ \
        --upload-file $1
```

Replace the asterisks in the `--user` parameter with the user password or API key.



The file `soh_text` functions similarly to `soh_mail`, and requires the user to configure a private SMS platform, or through a provider (e.g.: FireText). The soh_text file syntax is as follows:

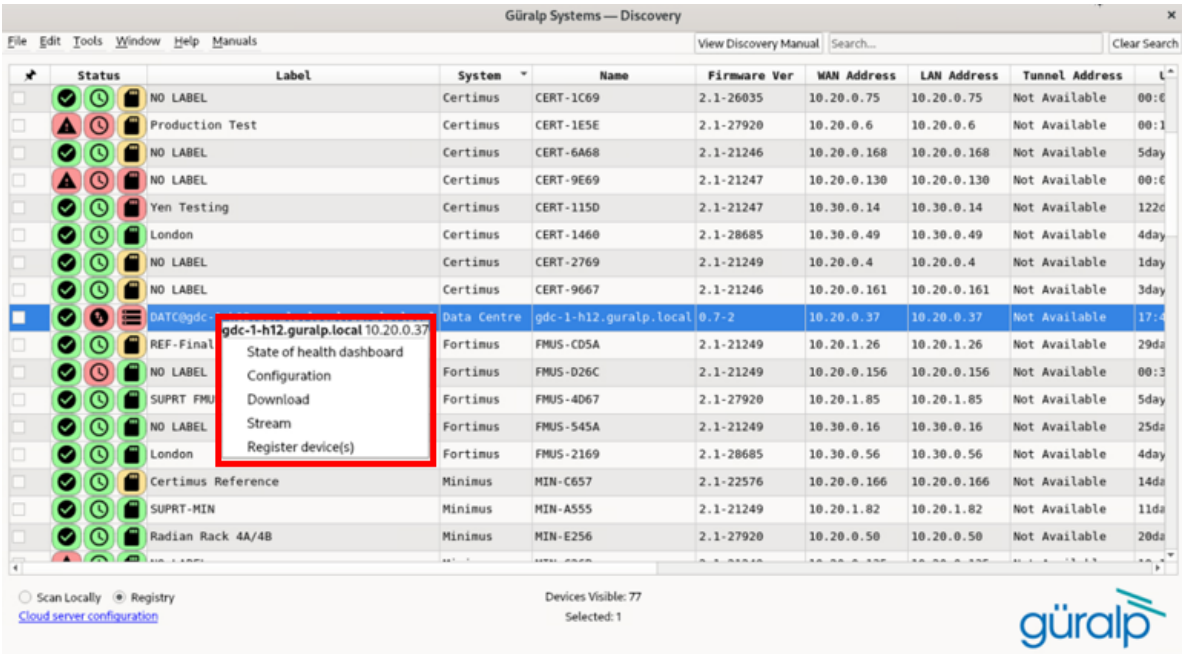
```
curl --url <api_url> --d apiKey=<api-key> --d to=<phone-number> --data-urlencode <message> -d from=<sender>
```

An example configuration is provided, using FireText API to send to a fake phone number with the entire command on one line:

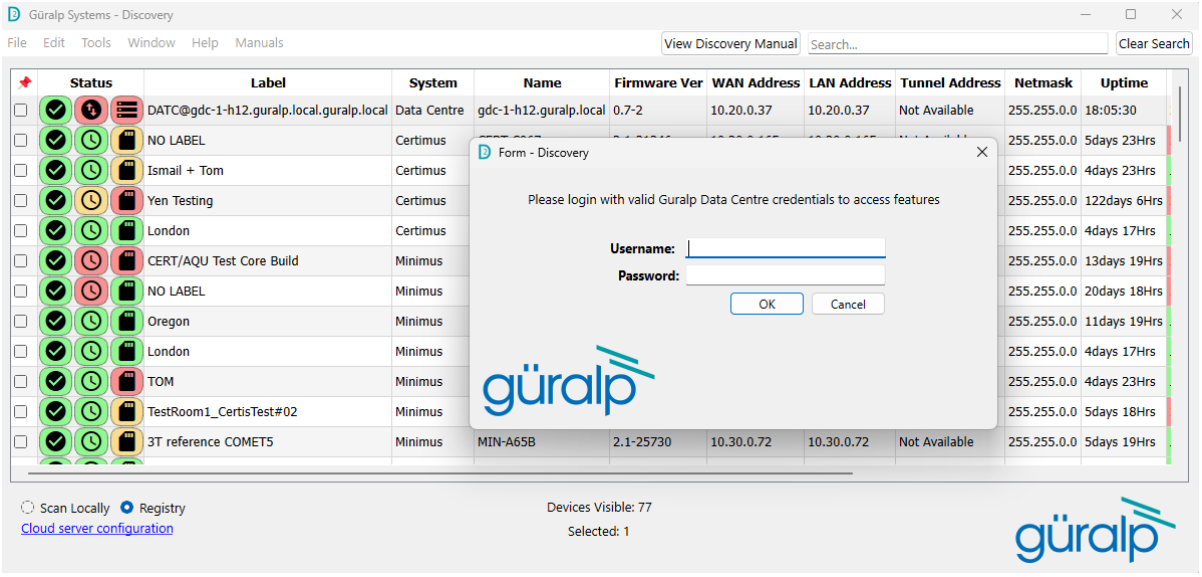
```
curl --url ‘https://www.firetext.co.uk/api/sendsms/json’ \
        -d apiKey=’**********’ \
        -d to=’01234567890’ \
        --data-urlencode ‘message=$(cat “$1”)’ \
        -d from= ‘Guralp’
```

4.1.3. Login and Device Registration

By right-clicking on your GDC instance, you can access features such as Stream and Download.



However first you need to register your Güralp Systems devices to your Güralp Data Centre (GDC) account. To do this, click 'Register devices(s)'. Now you will be prompted to login to your GDC account using the username and password provided by Güralp Systems. If you have forgotten your login credentials or if you would like to set up a GDC account with us, please contact support@guralp.com.

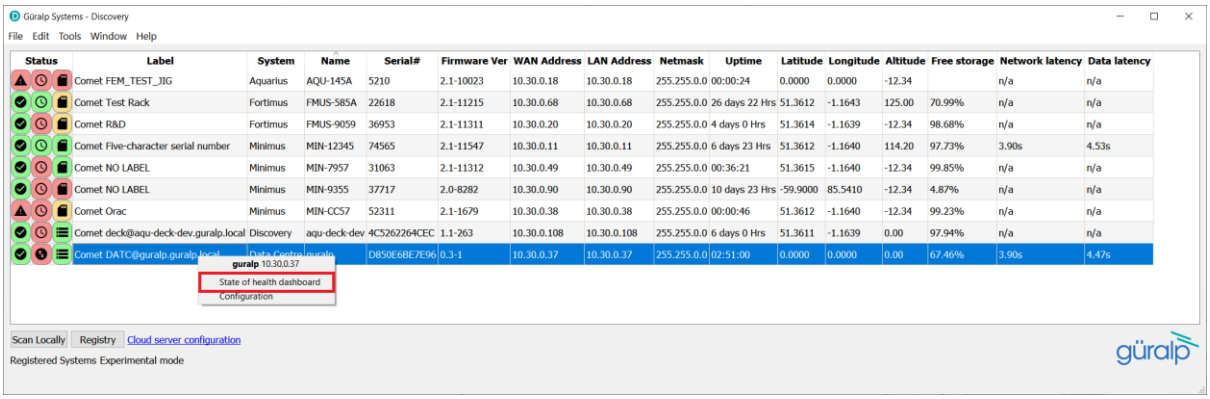


Once logged in you can add your devices to your GDC account following the instructions inside the widget.

Once you have logged in and registered devices to your GDC account you will be able to use your devices with the following features.

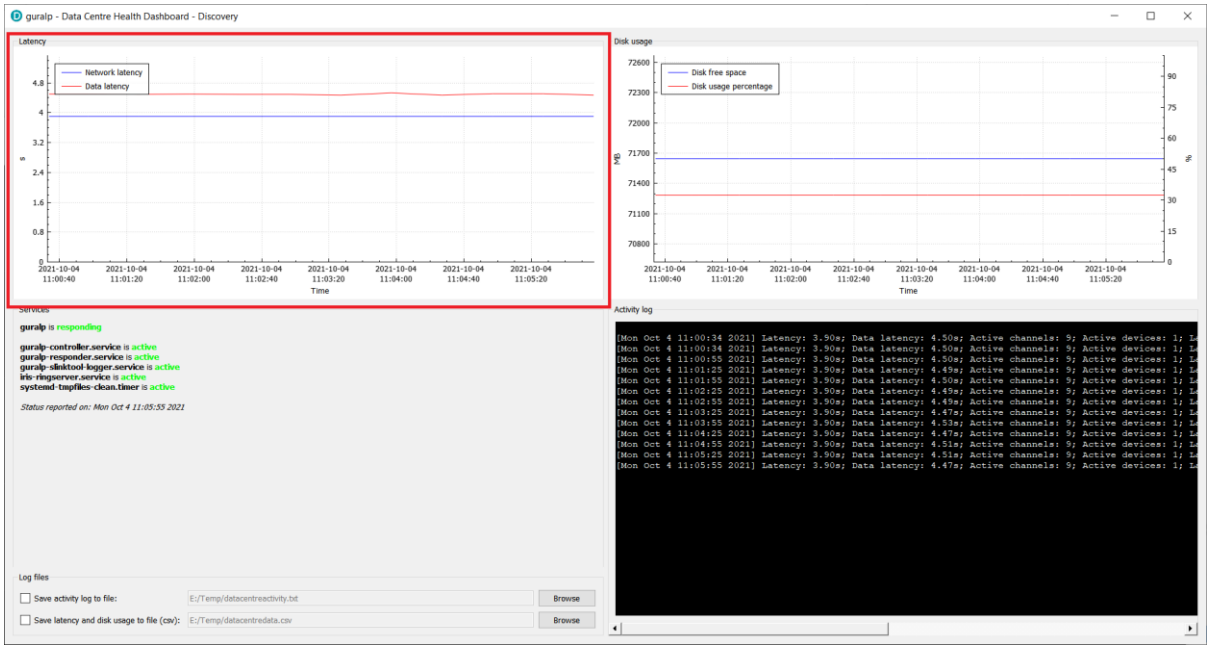
4.1.4. State of Health Dashboard

Data Centre state of health dashboard widget can be accessed through right-click menu of selected instance:

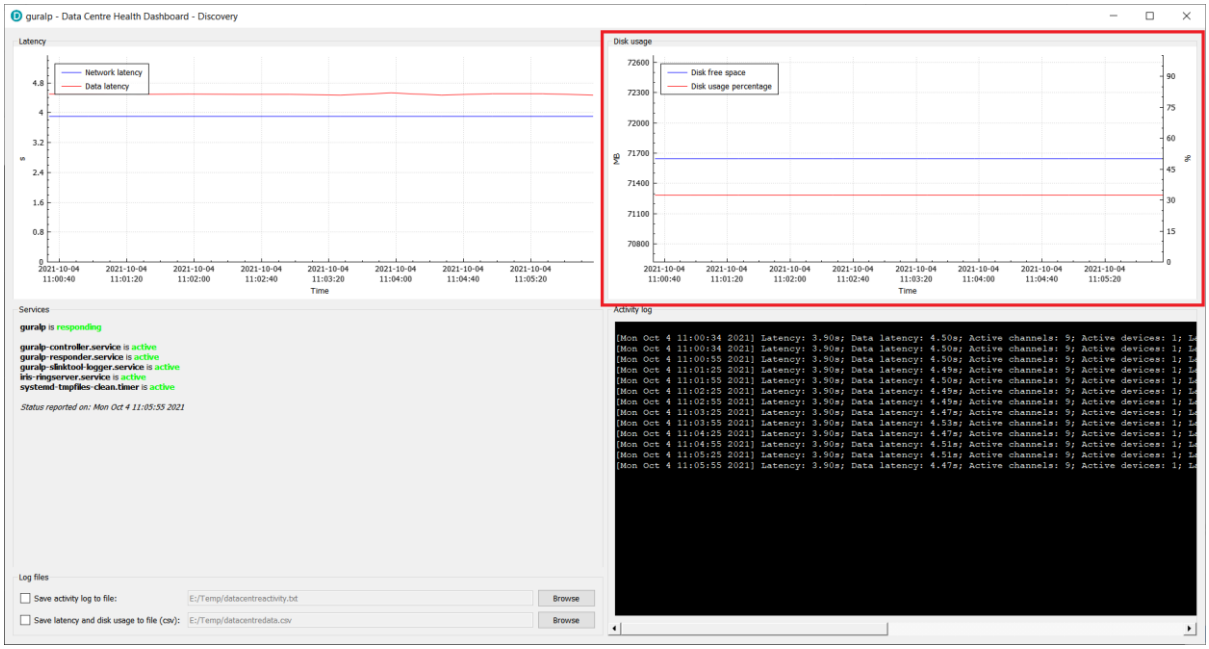


The state of health dashboard widget is divided into 4 main parts:

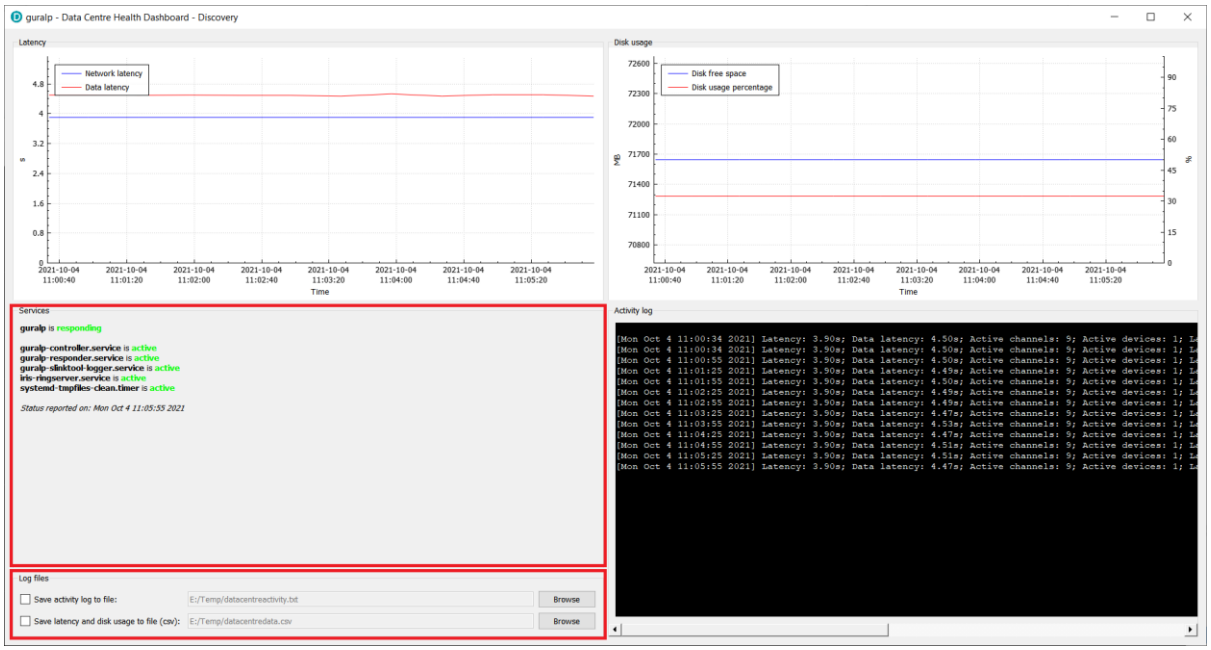
1. Top-left widget is a latency graph displaying the highest historical latency value for up to last 30 minutes.



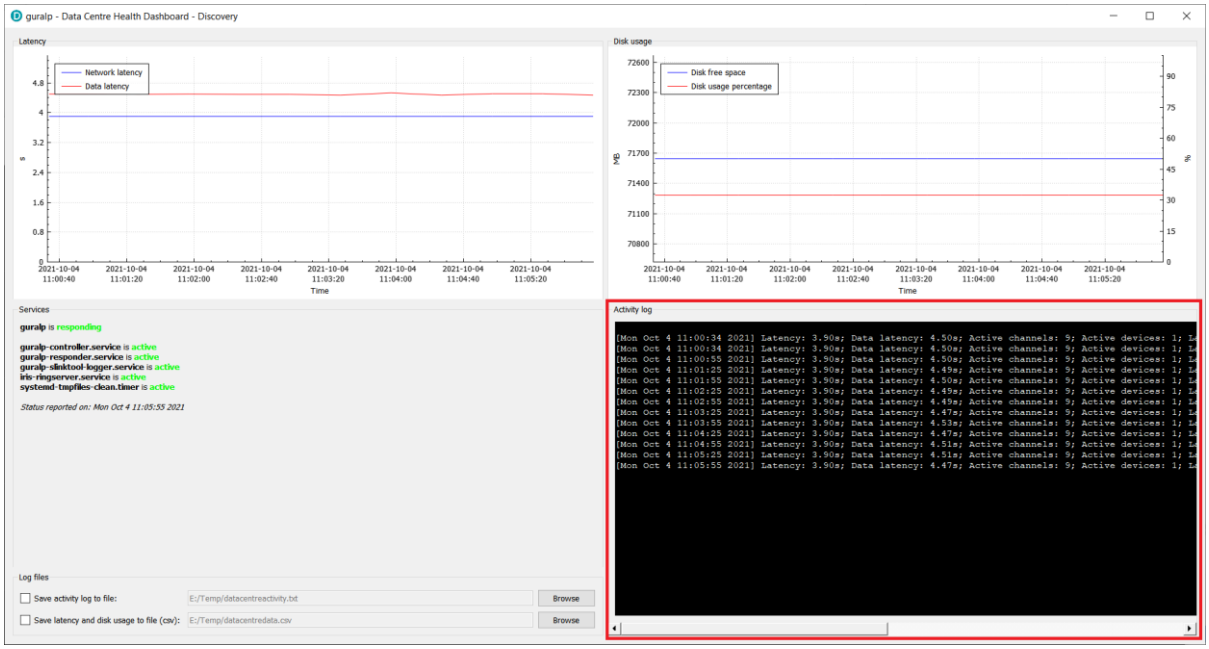
2. Top-right widget is a disk usage graph displaying the disk free space in MB (blue graph, left y axis) and disk used space percentage (red graph, right y axis)



3. Bottom-left widget displays the state of services running on the Data Centre and allows to configure the logfile and output data file for activity log and latency/disk usage data respectively



4. Bottom-right widget displays the activity log based on state of health information packets received



Log line contains the following information:

- Timestamp
- Latency value
- Sample latency value
- Number of active channels

- Number of active devices
- Latest sample timestamp
- Available disk space in KB
- Disk used space in KB
- Percentage value of free disk space

And is logged as single line in the following format:

```
{[Timestamp]} Latency: {Latency value}s; Sample latency: {Sample latency value}s; Active channels: {Number of active channels}; Active devices: {Number of active devices}; Latest sample timestamp: {Latest sample timestamp}; Disk available: {Available disk space in KB}; Disk used: {Disk used space in KB}; Disk free: {Percentage value of free disk space}%;
```

Example:

```
[Fri Jul 9 13:50:34 2021] Latency: 1.50s; Sample latency: 1.81s; Active channels: 26; Active devices: 2; Latest sample timestamp: Fri Jul 9 13:49:43 2021; Disk available: 73364480; Disk used: 54698232; Disk free: 25.44%;
```

4.1.5. Configuration

The configuration widget is split in 3 parts:

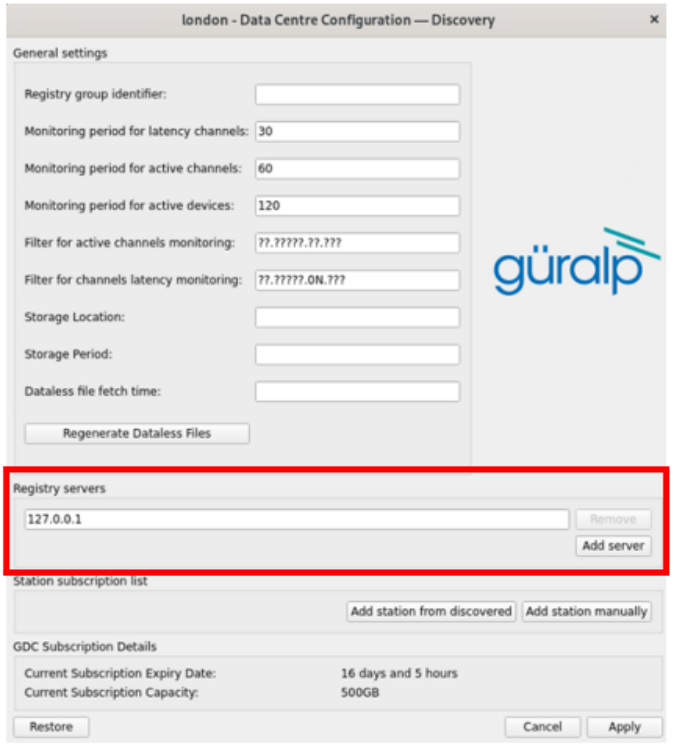
1. **General settings**

This section provides a set of text edit fields for Güralp Data Centre monitor:

- Registry group identifier – Group identifier used for registration to the Güralp responder server.
- Monitoring period for latency channels – Period of time in seconds that should be used for detecting the highest channel latency value.
- Monitoring period for active channels – Period of time in seconds that should be used for detecting the number of active channels.
- Monitoring period for active devices – Period of time in seconds that should be used for detecting the number of active devices.
- Filter for active channels monitoring – A SEED globing pattern defining the channels that are monitored for being active. Accepts a space (‘ ‘) separated list.
- Filter for channels latency monitoring – A SEED globing pattern defining the channels that are monitored for latency calculation. Accepts a space (‘ ‘) separated list.
- Storage information base location – Location that should be used for disk space information gathering. Leave blank for default (the service working directory).

Also, in this section you can request that GDC regenerates the dataless files for each connected station. Dataless files contain device metadata. By default, GDC will regenerate dataless files for all stations at 3 am every night and stored them at `/var/cache/guralp/dataless/`. This can be re-configured by re-defining the timer `/etc/system/system/guralp-dataless.timer`. GDC is also configured to request the dataless file from a device upon connecting it to a GDC. Therefore, GDC’s should always have dataless metadata available for all stations. The Regenerate Dataless Files button seen below can be used to manually regenerate dataless files for all connected stations if required at any time.

2. **Registry servers**



This section contains a list of Güralp responder server addresses that the Data Centre should notify its state of health to. At least one entry connecting to local loopback address (local instance of the responder service) should be configured. Additional servers can be added for redundancy and access extension by clicking on Add server button and providing the connection details.

3. **Station subscription list**

This section defines a list of all of the stations and channels that the Data Centre should connect to. It is particularly useful as it allows you to add and remove stations from your GDC instance (by enabling and disabling slink2dali and slinktool services).

In the configuration widget, if not preloaded with configuration, click on Restore button to retrieve the Data Centre configuration.

The widget populates the list of currently configured connections and provides a button to Remove the connection if no longer required. Connection subscription channel list can contain multiple SEED channel names (LOCATION_CODE.CHANNEL_CODE) defined as a space (‘ ‘) separated list.

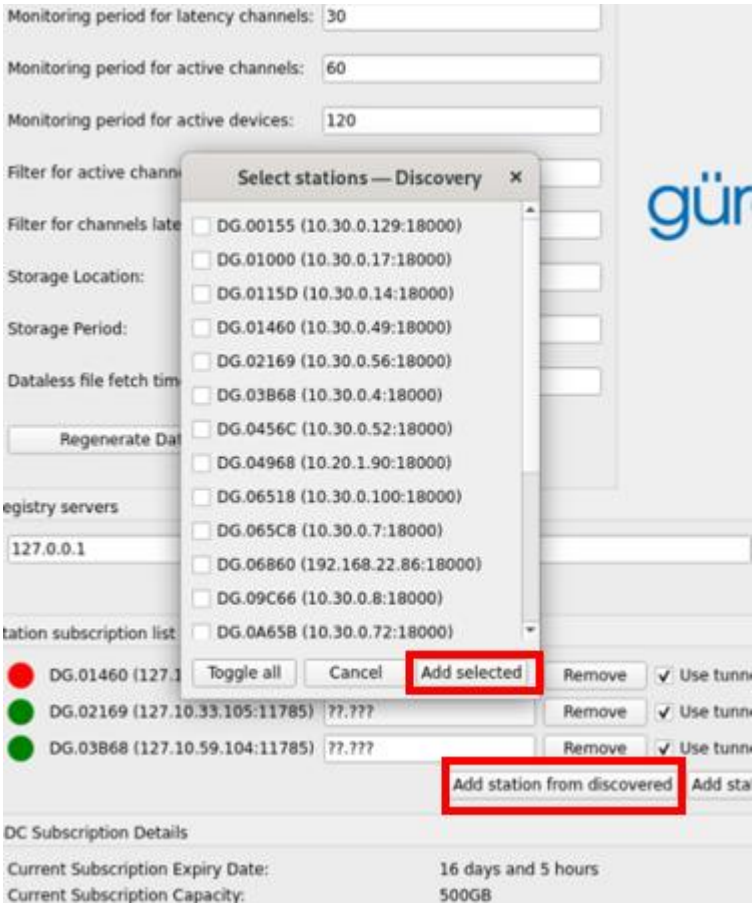

london - Data Centre Configuration — Discovery
General settings
Registry group identifier:
Monitoring period for latency channels: 30
Monitoring period for active channels: 60
Monitoring period for active devices: 120
Filter for active channels monitoring: ??.?????.??.???
Filter for channels latency monitoring: ??.?????.0N.???
Storage Location:
Storage Period:
Dataless file fetch time:
Regenerate Dataless Files
güralp
Registry servers
127.0.0.1
Remove
Add server
Station subscription list
DG.01460 (127.10.20.96:11785) ??.??? Remove Use tunnel connection
DG.02169 (127.10.33.105:11785) ??.??? Remove Use tunnel connection
DG.03B68 (127.10.59.104:11785) ??.??? Remove Use tunnel connection
Add station from discovered
Add station manually
GDC Subscription Details
Current Subscription Expiry Date: 16 days and 5 hours
Current Subscription Capacity: 500GB
Restore
Cancel
Apply

GDC supports multi-scale sensor networks with an optional failover mode, which allows the user to configure multiple connections to the same device to protect against system outages. The same device (identified via serial number) can be configured to communicate with the GDC through a variety of ways including cellular modem, local VPN, broadband WAN, satellite and wired WAN. When the currently active link fails due to some outage, Discovery will “fall-back” to a different link.

Connections can be quickly added from the list of stations populated in the Discovery desktop application main window through Add station from discovered button, or manually by clicking on Add station manually.

A. **Add station from discovered.**

When actioned, a new widget with a list of discovered stations is displayed and tick-boxes are available for selecting to which stations the Data Centre should connect to. Once list contains required stations, press Add selected button to confirm.



B. **Add station manually**

To add a device that is not supporting Güralp Discovery mechanism, this option opens a simple widget asking for details required to create the connection from the Data Centre to the device.

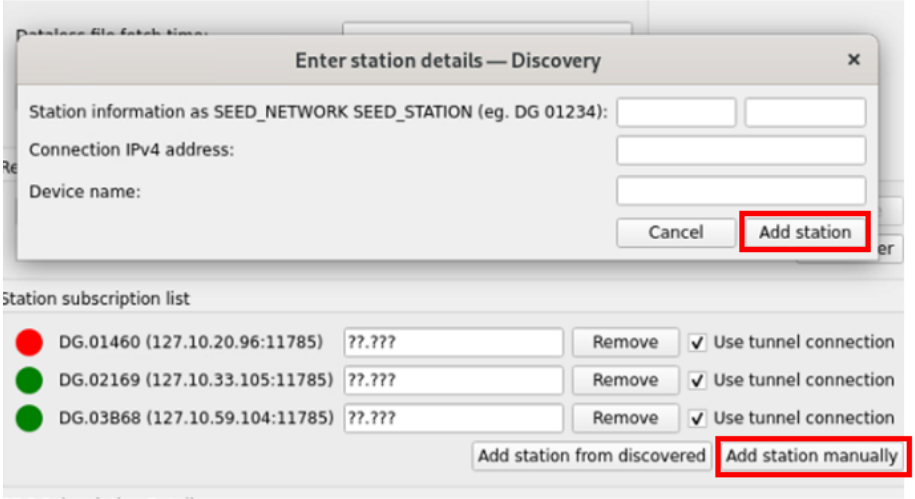
Required fields:

I. SEED Network code

II. SEED Station code

III. Connection IP address

To confirm, click on Add station button.



When all connections are added to the list, click Apply button to send the updated configuration to the Data Centre. Date Centre controller service will enable and start all required slinktool and slink2dali services. If station was removed from the list, the controller service will stop and disable the services used to connect to the removed device.

Manual service configuration can be performed but it is not recommended. In order to manually configure a connection, please log in to the Data Centre computer and use `systemctl` command to enable/disable and/or start/stop the slink2dali service. Slink2dali service is run with a set of parameters:

```
slink2dali@NC@STATC@LOCHN@CONNECTION@PORT@STATEFILESAVEINTERVAL.service
```

where:

1. NC, is SEED network code,
2. STATC, is SEED station code,
3. LOCHN, is SEED location and channel codes
4. CONNECTION, is the connection IP address or hostname
5. PORT, is the connection port
6. STATEFILESAVEINTERVAL, defines interval for state file save (can be blank)

Wildcard character for SEED location and channel name can be used and is represented by '_' character. Also, a list of location and channel names can be provided to a given service and should be separated with '-' character.

Example:

```
slink2dali@DG@0585A@_____@10.30.0.123@18000.service
```

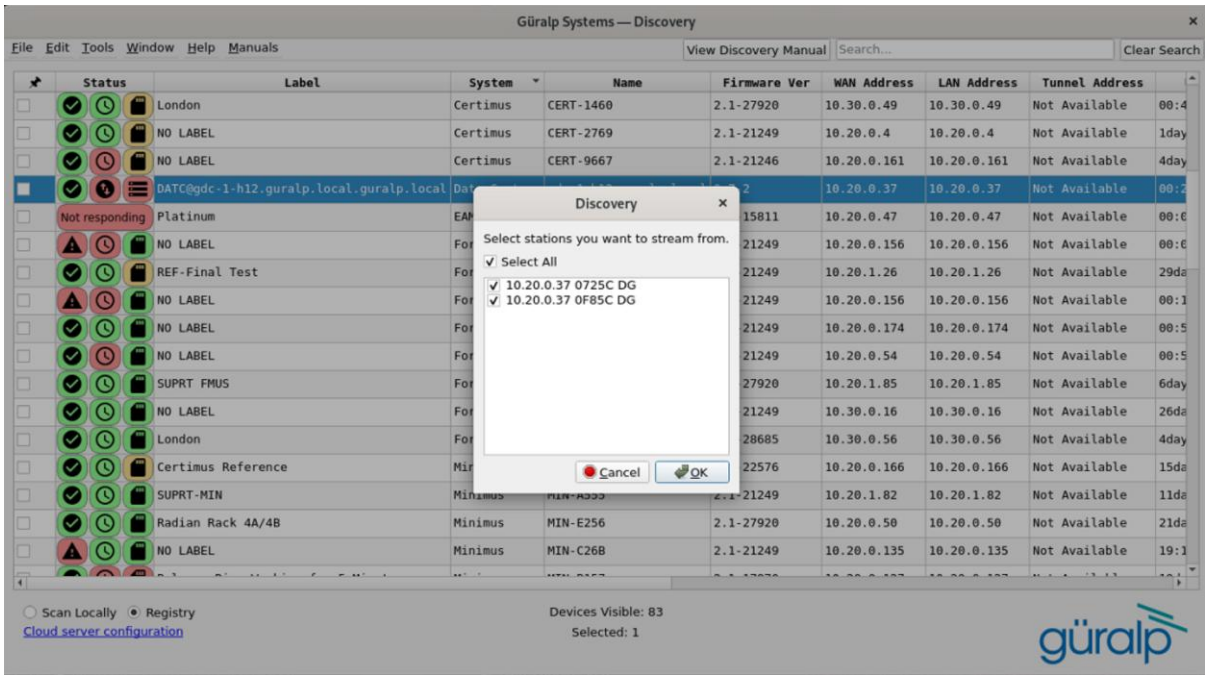
will connect to station 0585A of DG network using IP address 10.30.0.123 and port 18000 subscribing to any channel (wildcard selector of 5x '_' character).

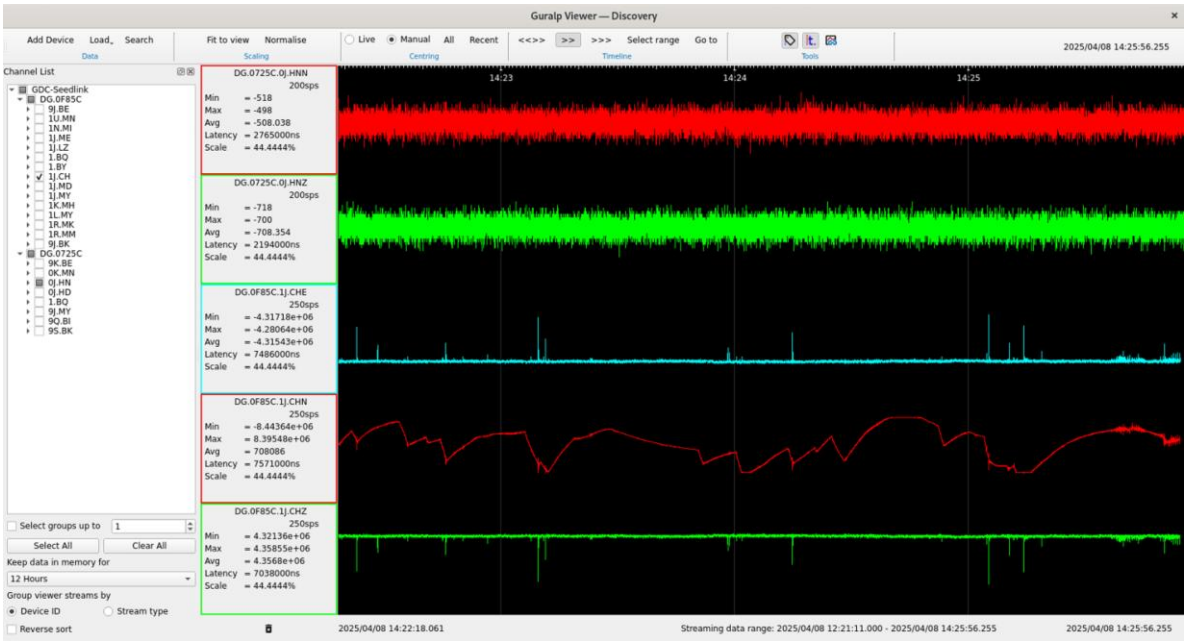
```
slink2dali@DG@0585A@0NHHZ-0NHHN-0NHHE@10.30.0.123@18000.service
```

will connect to station 0585A of DG network using IP address 10.30.0.123 and port 18000 subscribing to 0N.HHZ, 0N.HHN and 0N.HHE channels.

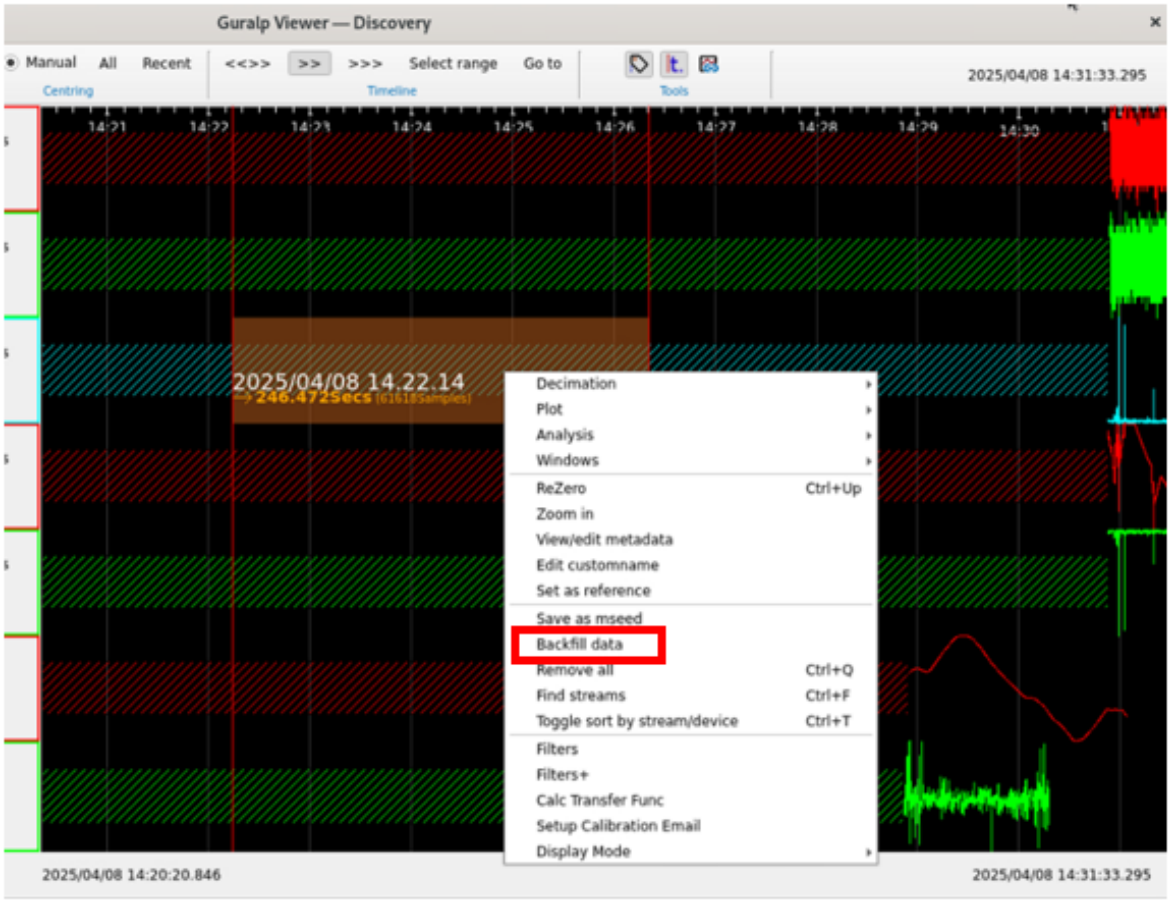
4.1.6. Stream and Backfill

You can live stream the miniSEED data coming out of the Iris ringserver via the 'Stream' function. Discovery will prompt you to select which stations on the GDC you want to stream from.

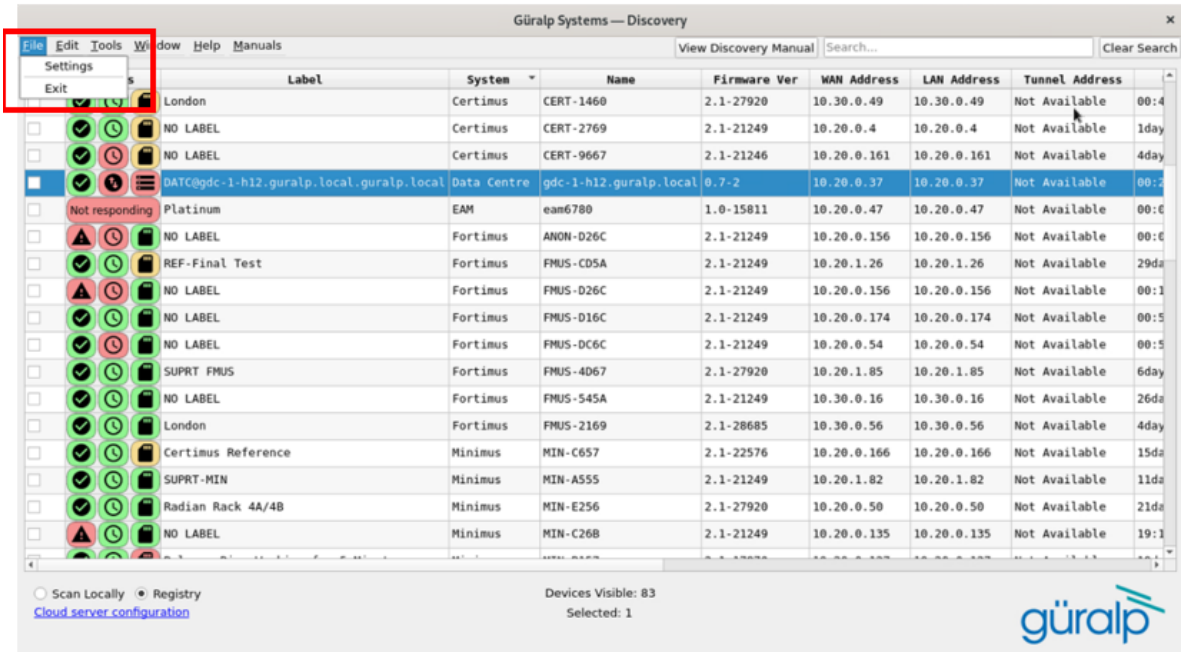




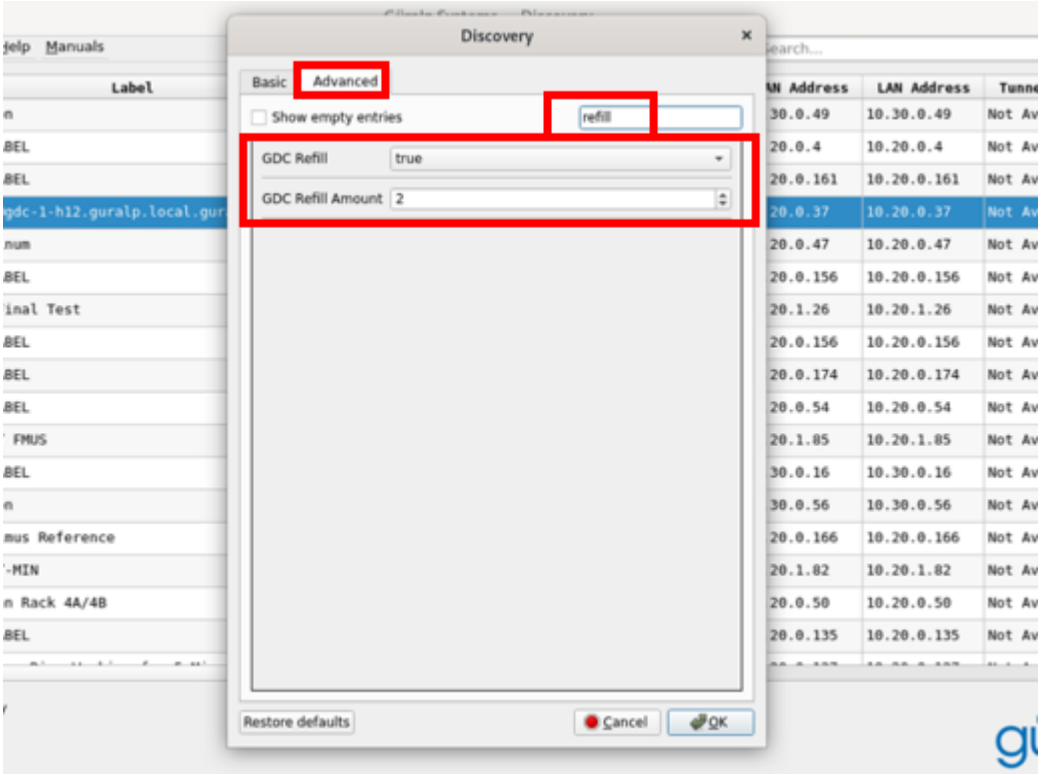
By right-clicking, you can backfill historical data available on the Data Centre.



Discovery can also be configured to automatically ‘refill’ a certain number of hours of data when you open a stream. To do this, go to advanced settings.

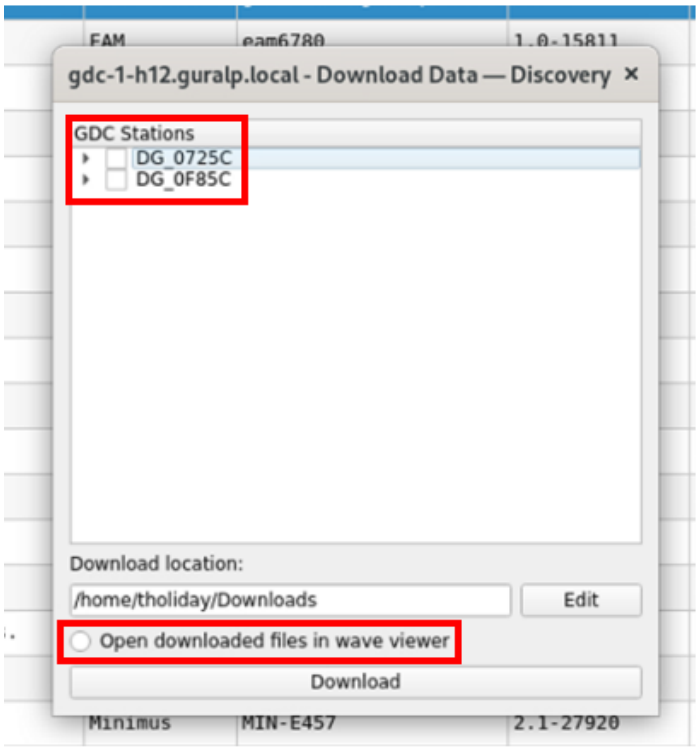


In advanced setting turn 'GDC Refill' to true and choose the number of hours you want Discovery to refill.



4.1.7. Download MiniSEED Files

Data can be downloaded via the 'Download' widget. Select which .mseed files to download from the list of stations and specify download location on your local machine. There is an option to open the downloaded files directly into a wave view window to be able to analyse the data and perform analysis on it.


gdc-1-h12.guralp.local - Download Data — Discovery
GDC Stations
DG_0725C
DG_0F85C
Download location:
/home/tholiday/Downloads
Edit
Open downloaded files in wave viewer
Download
Minimus
MIN-E457
2.1-27920

5. Configuration

5.1. Ringserver

IRIS ringserver does not require any specific configuration unless change in either default storage location (`/var/cache/guralp/miniseed/`) or time for which the data is kept on the hard drive is required. Application runs as an operating system service and listens for SeedLink connections on TCP port 18000. If required listening port is different to 18000 please contact support@guralp.com for instructions.

5.1.1. MiniSEED Files Storage Location Configuration

IRIS ringserver storage location is configured by the home directory of ringserver user and if required can be changed by system administrator by editing the user properties. Newly selected directory should have sufficient permission and ownership. The steps below show how to modify the home directory of the ringserver user in Red Hat Linux environment (please note that commands may require elevated permissions).

1. Copy `guralp-miniseed.conf` file to `/etc` system location

```
cp /usr/lib/tmpfiles.d/guralp-miniseed.conf /etc/tmpfiles.d/guralp-miniseed.conf
```

2. Edit the copied configuration file and append a new line specifying the new storage location:

```
d  /run/guralp/etc                              0755 root          root
F  /run/guralp/etc/iris-ringserver.conf         0640 root          ringserver
-   MSeedWrite %%n_%%s_%%l_%%c_%%Y_%%j.mseed
d  /var/cache/guralp/miniseed                   0775 root          ringserver
3d
d  /mnt/new/storage/directory                   0775 root          ringserver
3d
```

3. Save the changes made to the file
4. Modify the ringserver user home directory

```
usermod -d /mnt/new/storage/directory ringserver
```

5. Reboot the system to apply the changes

Note: only the following top directories can be used: `/home`, `/media`, `/mnt`, `/opt`, `/srv`, and `/var`.

5.1.2. MiniSEED Files Storage Auto-Clean Configuration

IRIS ringserver storage is controlled by system tmpfiles clean timer and can be configured by editing guralp-miniseed.conf file. In order to change the configuration of time that historical data is kept in the system please follow the steps below.

1. Copy tmpfiles.d guralp-miniseed.conf file to /etc system location

```
cp /usr/lib/tmpfiles.d/guralp-miniseed.conf /etc/tmpfiles.d/guralp-miniseed.conf
```

2. Edit the copied configuration file and change the age of storage location to the required value.

```
d  /run/guralp/etc                              0755 root          root
F  /run/guralp/etc/iris-ringserver.conf         0640 root          ringserver
-   MSeedWrite %%n_%%s_%%l_%%c_%%Y_%%j.mseed
d  /var/cache/guralp/miniseed                   0775 root          ringserver
3d
```

where `3d` is the age of temporary files (read more: https://www.freedesktop.org/software/systemd/man/tmpfiles.d.html).

3. Save and reboot the system

5.2. Slinktool

Slinktool process runs as a service with parameters provided through the service name in a form of a '@' separated list of values. Under normal operation there is no requirement to manually configure (enable/disable or start/stop) the service, this task is performed by the Data Centre controller on remote request from the Discovery desktop application.

Manual service configuration can be performed but it is not recommended. In order to do that, please log in to the Data Centre computer and use systemctl command to enable/disable and/or start/stop the slinktool service. Slinktool service is run with a set of parameters:

```
slinktool@NC@STATC@LOCHN@CONNECTION@PORT.service
```

where:

1. NC, is SEED network code,
2. STATC, is SEED station code,
3. LOCHN, is SEED location and channel codes
4. CONNECTION, is the connection IP address or hostname, for latency monitoring in the Data Centre this is set to 127.0.0.1 (localhost)
5. PORT, is the connection port

Wildcard character for SEED location and channel name can be used and is represented by '_' character. Also, a list of location and channel names can be provided to a given service and should be separated with '-' character.

Example:

```
slinktool@DG@0585A@_____@127.0.0.1@18000.service
```

will connect to station 0585A of DG network, subscribing to any channel (wildcard selector of 5x '_' character).

```
slinktool@DG@0585A@0NHHZ-0NHHN-0NHHE@127.0.0.1@18000.service
```

will connect to station 0585A of DG network, subscribing to 0N.HHZ, 0N.HHN and 0N.HHE channels.

5.3. Güralp Data Centre Controller Service

Güralp Data Centre controller does not require any specific configuration. Application runs as a service and does not need particular maintenance.

5.4. Güralp Data Centre Monitor Service

Güralp Data Centre Monitor service should be configured remotely by using the Configuration widget in the Discovery application. The configuration widget is available under right-click menu of a Data Centre row.

Manual configuration can be performed but is not recommended. Configuration is stored in a configuration file located in `/var/cache/guralp/guralp-monitor.ini` and contains a set of key-value entries:

Key	Description	Type
registry_addresses	Comma separated IP addresses of Güralp responder servers to which the state of health packet should be send to	Comma separated list of strings
registry_group_id	Güralp responder server group identifier string used. Please use “guralp3” as a default value	String
filter_monitored_channels	SEED globing style filter for channels activity monitoring	String
filter_monitored_latency_channels	SEED globing style filter for channels latency monitoring	String
monitoring_period_latency	Period of time in seconds that should be used to find the highest data latency	Integer
monitoring_period_active_channels	Period of time in seconds that should be used to detect number of active channels	Integer
monitoring_period_active_devices	Period of time in seconds that should be used to	Integer

	detect number of active devices	
storage_monitor_dir	Directory that should be used for storage monitoring, if this entry is not present, iris ringserver working directory is used.	String

Example `guralp-monitor.ini file` content:

```
[Version_1]
filter_monitored_channels="^.{1,2}\\..{1,5}\\..N\\..{1,3}"
filter_monitored_latency_channels="^.{1,2}\\..{1,5}\\..N\\..{1,3}"
monitoring_period_active_channels=120
monitoring_period_active_devices=300
monitoring_period_latency=30
registry_addresses=127.0.0.1,
registry_group_id=guralp3
storage_monitor_dir=/var/cache/guralp/miniseed
```

5.5. Güralp Responder Service

Güralp responder service provides a configuration option for specifying a wildcard group identifier that grants access to all of the registered devices. Wildcard group identifier can be used in the Discovery desktop application to list all of the stations that are connected to the Data Centre even if they are registering with a different, non-matching, group identifiers.

Wildcard group identifier can be set by creating a configuration file under tmpfiles.d directory. The configuration file generates the content of `/run/guralp/etc/wildcard_groupid.txt` file that is read by the responder service on startup and the content is used as the wildcard. To configure a custom wildcard string please follow the steps below.

1. Copy tmpfiles.d configuration file to system location.

```
cp /usr/lib/tmpfiles.d/guralp-responder.conf /etc/tmpfiles.d/guralp-responder.conf
```

2. Edit the configuration file `/etc/tmpfiles.d/guralp-responder.conf` to generate the selected wildcard string. Example content of `/etc/tmpfiles.d/guralp-responder.conf`

```
d  /run/guralp/etc                      0755 root          root
F  /run/guralp/etc/wildcard_groupid.txt 0640 root          guralpmonitor    -
customWildcard
```

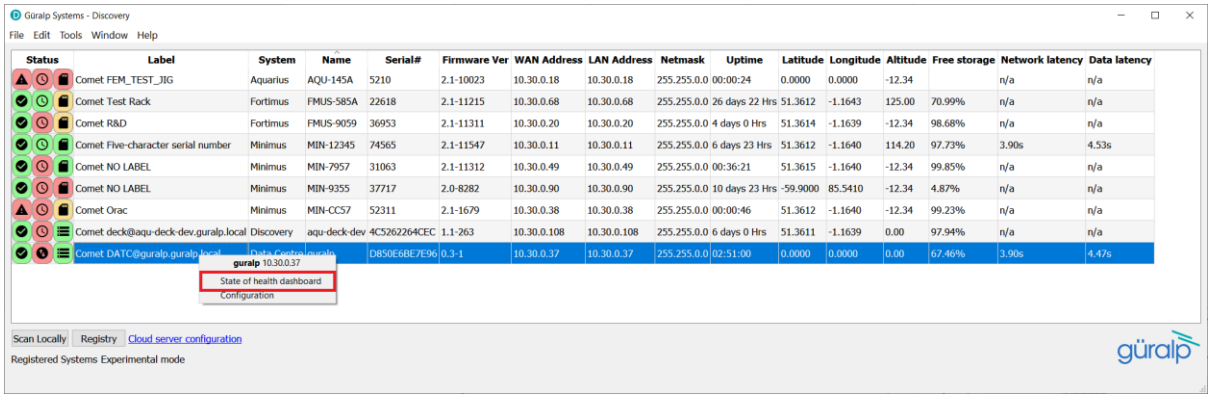
Where `customWildcard` is the new wildcard string.

3. Save the modified file.
4. Restart the operating system.

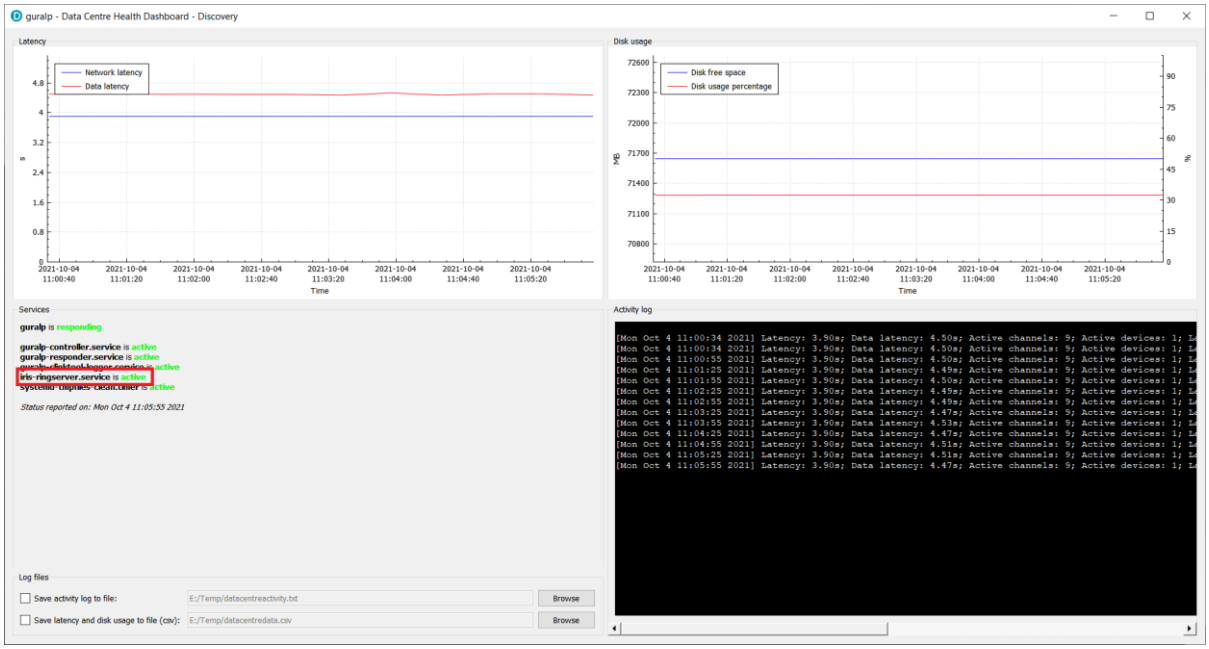
6. System Monitoring

6.1. IRIS Ringserver

IRIS ringserver service status is displayed in State of Health dashboard of the Data Centre instance and should be regularly monitored to assure correct operation of the software. The State of Health dashboard is available under right-click menu of the Data Centre row in the Discovery desktop application.



Bottom left part of the dashboard provides information about the state of services required for correct data centre operation. If for any reason service displayed as “iris-ringserver.service” is not listed as “active”, please log in to the Data Centre computer and investigate using systemctl tools. If problem persists, please contact Güralp support.



6.2. Slinktool

Slinktool service is controlled by the Data Centre controller service and does not require operational maintenance. In a rare case when the reported latency seems not

to be correct, it is recommended to login to the Data Centre computer and check the relevant slinktool service status using systemctl command.

6.3. Slink2dali

Slink2dali service is controlled by the Data Centre controller service and does not require operational maintenance. In a rare case when the data is not received on the client side, it is recommended to login to the Data Centre computer and check the relevant slink2dali service status using systemctl command.

Slinktool and slink2dali services can be remotely restarted by removing and re-adding the seismic station to the list of connections in the Data Centre configuration widget.

6.4. Güralp Data Centre Controller Service

Güralp Data Centre controller service does not require regular maintenance. If the service is not responding or configuration exchange between the Discovery desktop application and the Data Centre is not working, please ssh log in to the Data Centre computer and check the status of guralp-controller service:

```
systemctl status guralp-controller.service
```

If status returned is different from active:

```
guralp-controller.service - Güralp data centre controller
   Loaded: loaded (/usr/lib/systemd/system/guralp-controller.service;
enabled; vendor preset: disabled)
   Active: active (running) since Tue 2021-07-20 09:58:27 BST; 6min ago
 Main PID: 1660 (guralp-controll)
    Tasks: 1 (limit: 48584)
   Memory: 12.2M
   CGroup: /system.slice/guralp-controller.service
           1660 /usr/libexec/guralp-controller -exec

Jul 20 09:58:27 guralp systemd[1]: Started Güralp data centre controller.
Jul 20 09:58:30 guralp guralp-controller[1660]: Service initialised
Jul 20 09:58:30 guralp guralp-controller[1660]: Listening on port 11788
```

Restart the service using systemctl command:

```
systemctl restart guralp-controller.service
```

If problem persists, please contact Güralp Systems support team at support@guralp.com.

6.5. Güralp Data Centre Monitor Service

Güralp Data Centre monitor service sends periodic state of health information packet to all configured Responder servers and performs latency monitoring. If the Data Centre row in the Discovery desktop application is listed as non-responding it means that the application did not receive state of health information from the Data Centre for more than 90 seconds what may indicate that either the server is down, or the

Data Centre monitor is not working, or Güralp responder service stopped working, or there is no connection between the Discovery application and the server. In such situation, please try to ssh log in to the server, and if successful check the status of the Data Centre service by running systemctl command:

```
systemctl status guralp-monitor.service
```

The service should be in active state:

```
guralp-monitor.service - Güralp data centre monitor
   Loaded: loaded (/usr/lib/systemd/system/guralp-monitor.service; enabled;
vendor preset: disabled)
   Active: active (running) since Tue 2021-07-20 09:58:27 BST; 20min ago
 Main PID: 1659 (guralp-monitor)
    Tasks: 4 (limit: 48584)
   Memory: 21.2M
   CGroup: /system.slice/guralp-monitor.service
           1659 /usr/libexec/guralp-monitor -exec

Jul 20 09:58:27 guralp systemd[1]: Started Güralp data centre monitor.
```

If the service is not in an active state, please try to restart the service using systemctl command:

```
systemctl restart guralp-monitor.service
```

If problem persists, please contact Güralp Systems support team at support@guralp.com.

Güralp Data Centre monitor service periodically monitors the slinktool log files in order to gather accurate latency information and feed it to the Güralp responder service for further processing. The highest latency number detected in the configured latency monitoring window is set as Data Centre latency figure and is displayed in the Discovery desktop application. The service records all of the latency data information in a csv and log files stored under home directory of the “slinklat” operating system user, set by default to `/var/cache/guralp/latency`. Default configuration keeps the recorded data files for 10 days and older files are getting deleted through tmpfiles.d mechanism.

6.6. Güralp Responder Service

Güralp responder service is responsible for collecting and distributing state of health information from Güralp devices. Potential problems caused by malfunction in the service operation will cause the Discovery desktop application to display incorrect state of health information or the list of devices in the application main windows will be empty under the “Registry” mode. If the responder service is malfunctioning, please ssh log in to the Data Centre computer and verify the state of the service by executing the following systemctl command:

```
systemctl status guralp-responder.service
```

Service should be in active state:

```
guralp-responder.service - Güralp responder (a.k.a. Güralp registry server)
   Loaded: loaded (/usr/lib/systemd/system/guralp-responder.service;
```

```
enabled; vendor preset: disabled)
   Active: active (running) since Tue 2021-07-20 11:06:17 BST; 7min ago
 Main PID: 1639 (guralp-responder)
    Tasks: 5 (limit: 48584)
   Memory: 500.0K
   CGroup: /system.slice/guralp-responder.service
             1639 /usr/libexec/guralp-responder -d

Jul 20 11:06:17 guralp guralp-responder[1639]: Discovery Server version
0.80-0091
Jul 20 11:06:17 guralp guralp-responder[1639]: Devices expire after one day
Jul 20 11:06:17 guralp guralp-responder[1639]: Opening pinger UDP listener,
port 11788
Jul 20 11:06:17 guralp guralp-responder[1639]: Socket 4
```

If the service is not in an active state, please try to restart the service using systemctl command:

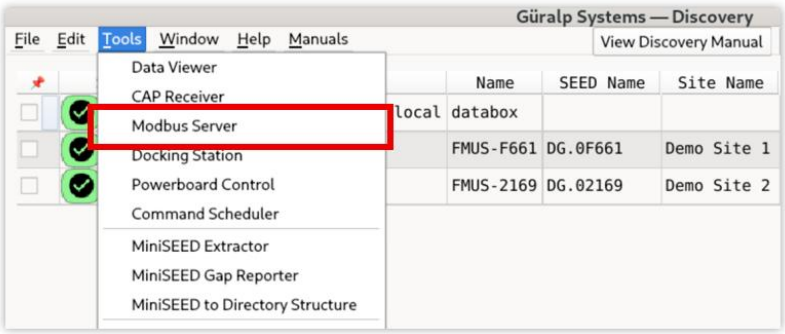
```
systemctl restart guralp-responder.service
```

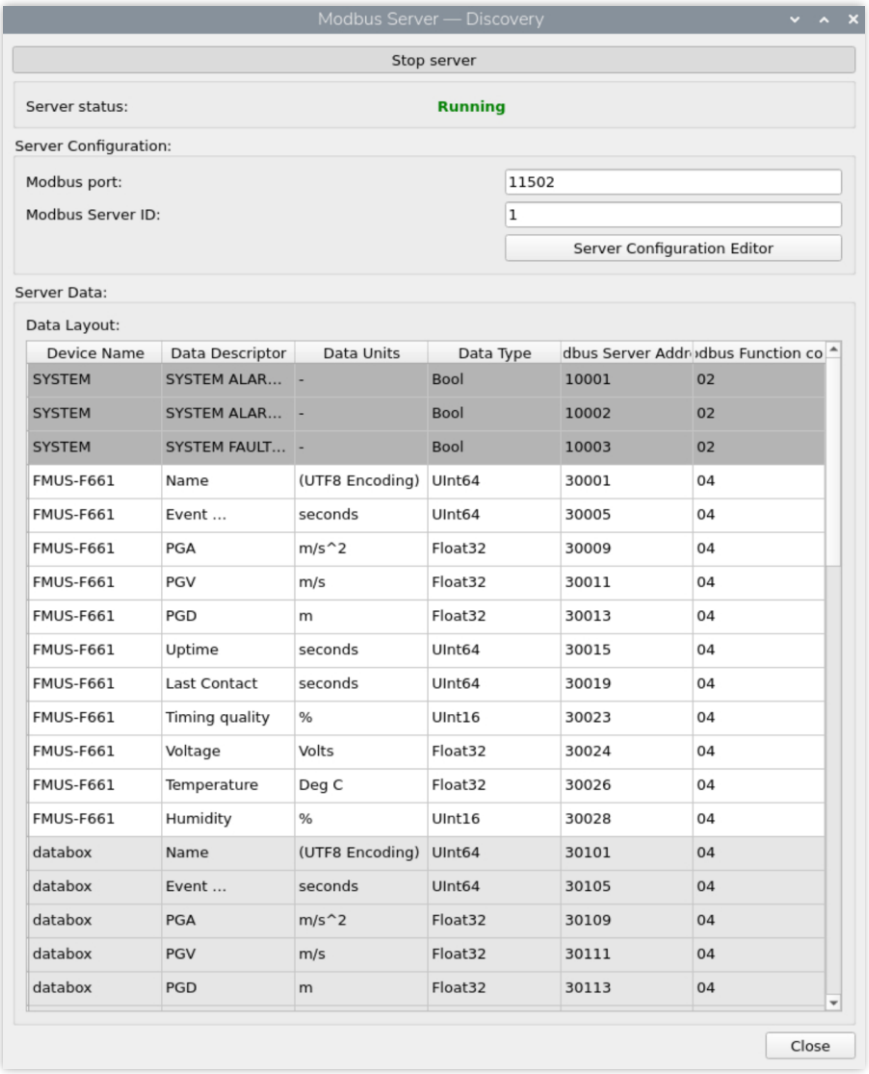
If problem persists, please contact Güralp Systems support team at support@guralp.com.

7. Güralp Discovery Modbus Server Tool

The Modbus Server tool within Güralp Discovery allows for a Modbus TCP Server to be hosted from a device running the Discovery software. This enables the communication of information from a Discovery program to an outside client via the Modbus Protocol. When run at startup, Discovery automatically begins hosting a Modbus server on the default port 11502 TCP. To access configuration, the Modbus Server widget is accessible from the Tools menu in the main interface.

Further information about the contents of the server can be found in Section 5.2 of the Discovery Software Manual.





Güralp Data Box contains a Python script which can be used to test communication with the Modbus server. This can be used via the terminal on the GDB. Open a terminal window and enter:

```
./query_modbus_server --ip [IP]
```

Where `[IP]` is the address of the server host. As the script is being run on the same device as the server is being hosted, the default localhost address 127.0.0.1 should be applicable for this testing procedure. The script takes additional arguments in the form of:

```
./query_modbus_server --ip [IP] --port [PORT] --slave [SLAVE]
```

Where `[PORT]` is the same port configured in the Modbus Server widget (default 11502) and `[SLAVE]` refers to the register being queried (default 1). The script can be found at the file address:

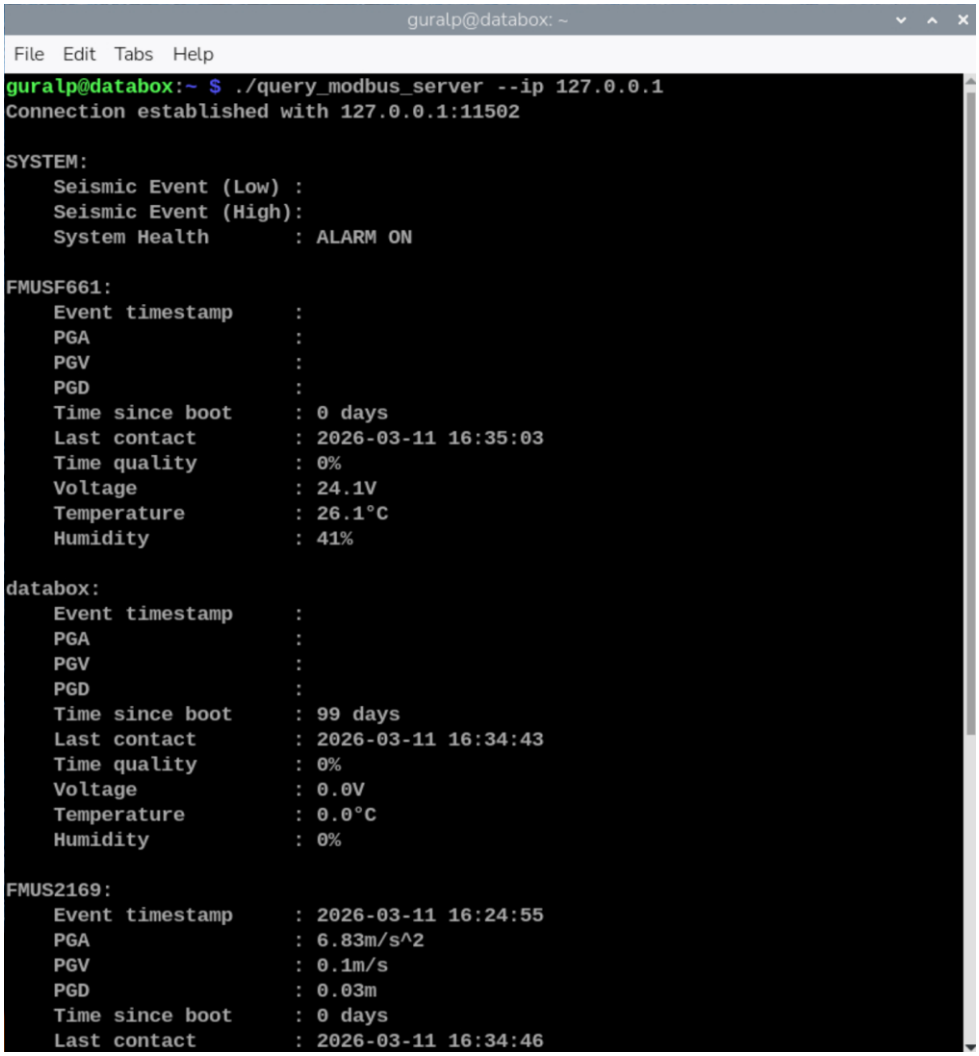
```
/home/guralp/query_modbus_server
```

In addition to `query_modbus_server` GDB contains a Python script named `monitor_modbus_server` which automatically queries the Modbus server at a fixed time interval.

`/home/guralp/monitor_modbus_server`

Usage of the script is identical to `query_modbus_server`, with one additional argument for query interval (in seconds).

`./monitor_modbus_server --ip [IP] --port [PORT] --slave [SLAVE] --interval [interval]`



8. Support

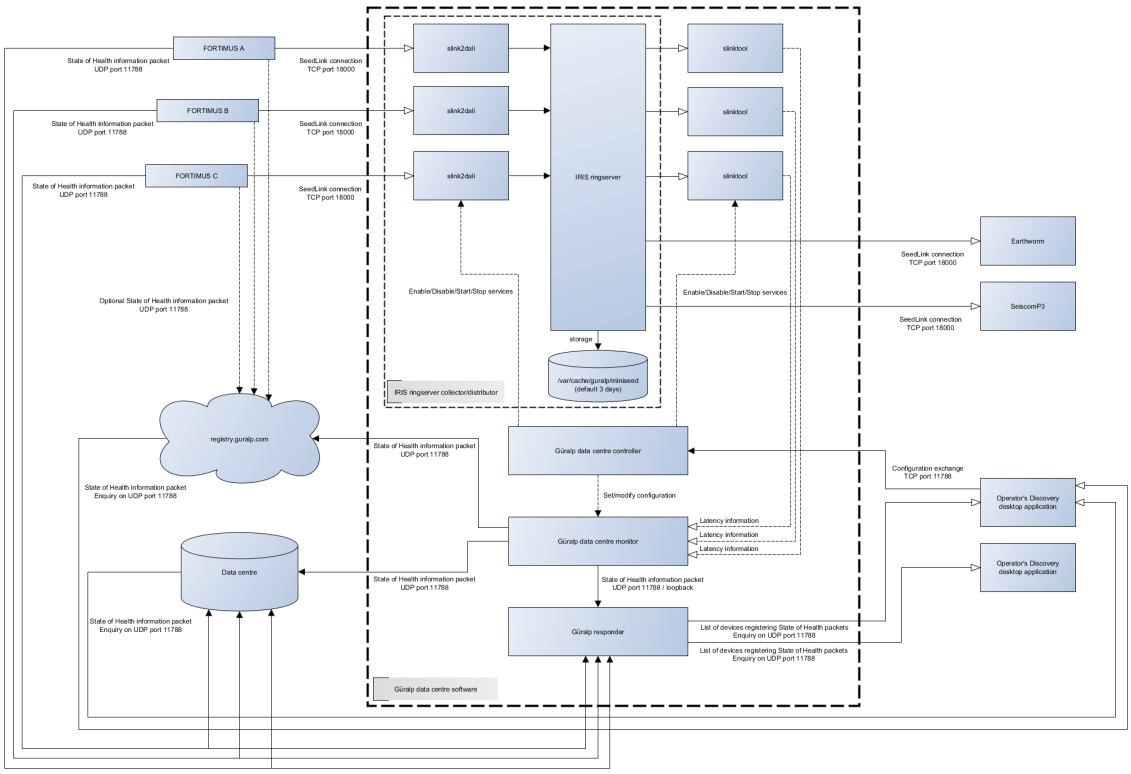
For support enquiries, please contact support@guralp.com.

Güralp Systems Limited
Midas House, Calleva Park, Aldermaston,
Reading, RG7 8EA,
United Kingdom

Tel: +44 118 981 9056
Fax: +44 118 981 9943

E-mail: sales@guralp.com

9. Appendix 1 - Architecture



9.1. Overview

Güralp Data Centre software consists of several applications providing system state of health monitoring, data collection and distribution, and remote configuration capabilities. This document describes the software packages provided, explaining the architecture, software components and communication between them.

9.2. Software Components

List of all applications included in Data Centre software package:

- IRIS ringserver
- slinktool
- Güralp Data Centre controller service
- Güralp Data Centre monitor
- Güralp responder service

9.2.1. Ringserver

Ringserver is a well-established stream-oriented packet ring buffer used primarily to transport packetized time series of data. Ringserver supports TCP based protocols:

DataLink, SeedLink, and HTTP/WebSocket. The program has a built-in miniSEED archiver and in default configuration provided by Güralp, keeps the data archived for last 3 days.

Configuration details can be found in Güralp Data Centre Operator Manual.

Read more: https://github.com/iris-edu/ringserver

9.2.2. Slinktool

Slinktool is used as a diagnostic SeedLink client for latency monitoring. The tool connects to the ringserver and examines the latency of the data packets received. Latency is reported to the system log which is then read by Güralp Data Centre Monitor application and the highest latency value is sent in the state of health packet.

Read more: https://github.com/iris-edu/slinktool

9.2.3. Güralp Data Centre Controller Service

Güralp Data Centre controller service is a stand-alone application, run as a service, responsible for SeedLink connection management and Güralp Data Centre Monitor configuration. The application communicates with the Discovery desktop application through TCP connection on port 11788 using proprietary protocol in both directions: Discovery-service and service-Discovery.

Service is enabling/disabling and starting/stopping slink2dali and slinktool services responsible for data collection and latency calculation for each seismic station connected to data centre. Additionally, this service modifies the configuration of the Data Centre Monitor with settings configured by the Operator in a dedicated graphical user interface widget in the Discovery desktop application.

9.2.4. Güralp Data Centre Monitor

Güralp Data Centre Monitor service is a stand-alone application, run as a service, responsible for the periodic sending of state of health packets containing the latest information about Data Centre. State of health packets are sent to selected Güralp responder instances and can be configured by the Operator through either a dedicated GUI widget in Discovery desktop application, or by manually editing the `guralp-monitor.ini configuration` file.

The configuration file is located in `/var/cache/guralp/guralp-monitor.ini` and contains pairs of key-value entries:

Key	Description	Type
registry_addresses	Comma separated IP addresses of Güralp responder servers to which the state of health packet should be send to	Comma separated list of strings

registry_group_id	Güralp responder server group identifier string used	String
filter_monitored_channels	SEED globing style filter for channels activity monitoring	String
filter_monitored_latency_channels	SEED globing style filter for channels latency monitoring	String
monitoring_period_latency	Period of time in seconds that should be used to find the highest data latency	Integer
monitoring_period_active_channels	Period of time in seconds that should be used to detect number of active channels	Integer
monitoring_period_active_devices	Period of time in seconds that should be used to detect number of active devices	Integer
storage_monitor_dir	Directory that should be used for storage monitoring, if this entry is not present, ringserver's working directory is used.	String

Example file:

```
[Version_1]
filter_monitored_channels="^.{1,2}\\..{1,5}\\..N\\..{1,3}"
filter_monitored_latency_channels="^.{1,2}\\..{1,5}\\..N\\..{1,3}"
monitoring_period_active_channels=120
monitoring_period_active_devices=300
monitoring_period_latency=30
registry_addresses=127.0.0.1
registry_group_id=guralp3
storage_monitor_dir=/var/cache/guralp/miniseed
```

Data Centre Monitor provides the following functionality:

1. It finds the highest latency for channels accepted by the filter and time period configured.

   The health monitor periodically reads the system log generated by slinktool to find the highest channel latency satisfied by the filter and time restrictions. The length of time over which to examine the log file in search of the highest latency is configured in `guralp- monitor.ini` file as `monitoring_period_latency` and is expressed in number of seconds. The channels to be considered for latency search are configured as `filter_monitored_latency_channels` as a SEED globing expression, for example: DG.?????.0L.??? will select all channels from network DG and location 0L (i.e.: DG.12345.0L.HHZ, DG.12345.0L.HHN, DG.54321.0L.CHZ).

2. It scans for a number of active channels in the time period configured.

   As for the latency, data centre monitor is periodically examining system log generated by slinktool to monitor the number of active channels that pass through the SEED globing filter configured in `filter_monitored_channels` entry of `guralp- monitor.ini` file. System log is scanned for a period as configured in `monitoring_period_active_channels` entry.

3. It scans for a number of active devices in the time period configured.

   Similar to active channels monitoring functionality but does not provide filter configuration. The log file is examined for a period as configured in `monitoring_period_active_devices` entry of `guralp-monitor.ini` file.

4. It monitors the state of important services.

   Health monitor periodically checks the state of services required for the Data Centre to operate correctly. The list of services is configured in `guralp-monitor.ini` file under `monitor_service` entry but it is highly recommended not to modify this configuration entry.

5. It sends state of health information to Güralp Responder instances.

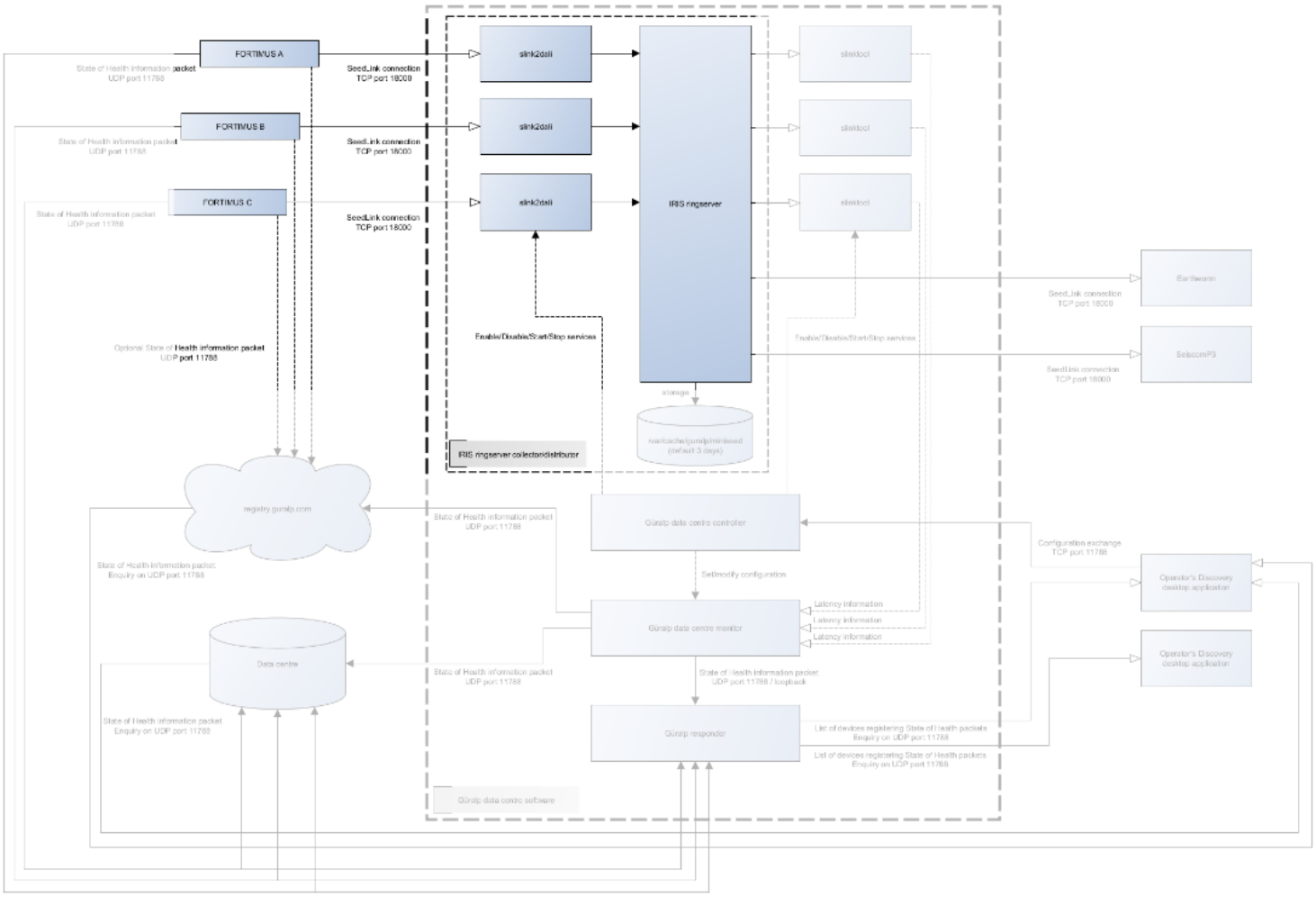
   The service to notify receivers about the latest state of health of the Data Centre sends periodic UDP packets on port 11788 to all configured instances of the Güralp Responder servers. The list of servers is configured in `guralp-monitor.ini` file under `registry_addresses` entry. Packets are sent with group identifier configured as `registry_group_id` value.

9.2.5. Güralp Responder Service

Güralp responder service is a stand-alone application, run as a service, responsible for collection and re-distribution of state of health information packets sent by Güralp seismic stations, Data Centre Monitors, and in special cases, Discovery desktop applications. Responder service is listening on UDP port 11788 for incoming state of health packets (device registration), and state of health enquiry (device state of health request).

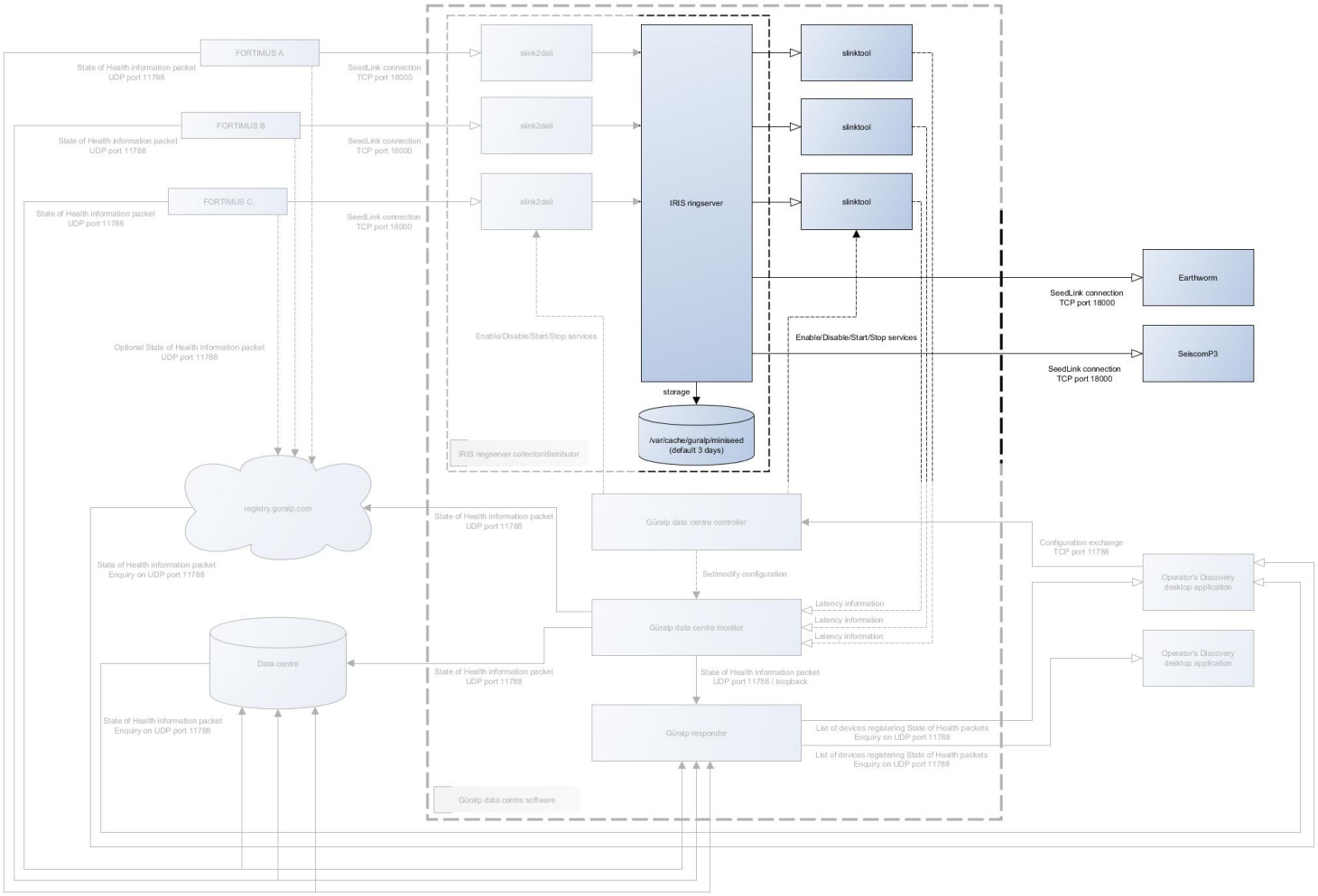
9.3. Communication Overview

9.3.1. Data Collection



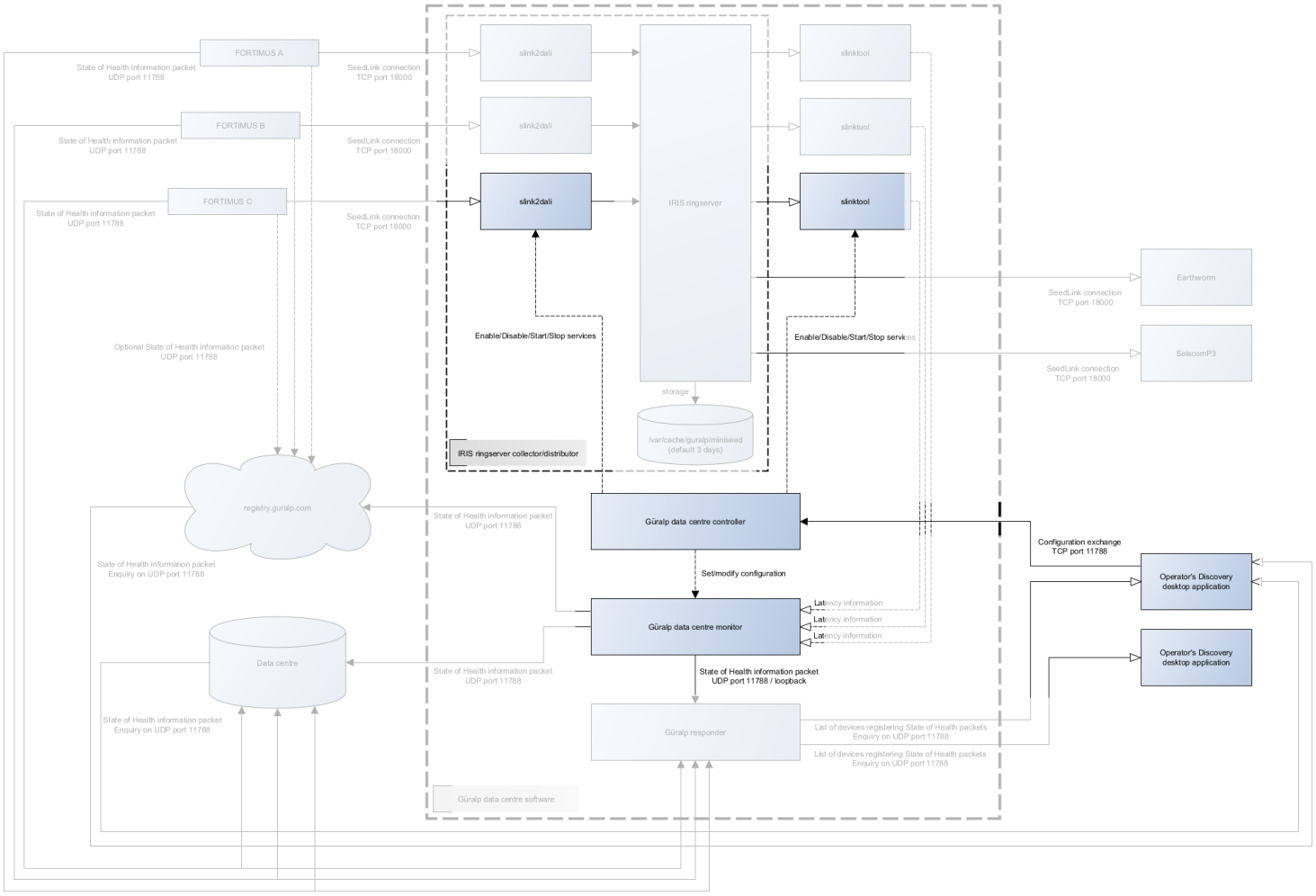
IRIS ringserver uses slink2dali service to collect the data from the seismic station. Data is collected using SeedLink protocol through TCP connection on port 18000. Data acquisition for a given station can be started either remotely through Discovery desktop application, or manually by enabling/starting slink2dali service for the station when logged into the Data Centre computer, more details on how to start a connection can be found in Güralp Data Centre Operator Manual.

9.3.2. Data Distribution



Data Centre data distribution is handled by IRIS ringserver and is provided as SeedLink and/or DataLink connection instantiated by the remote client on TCP link. Default port configuration is 18000 for SeedLink and 16000 for DataLink.

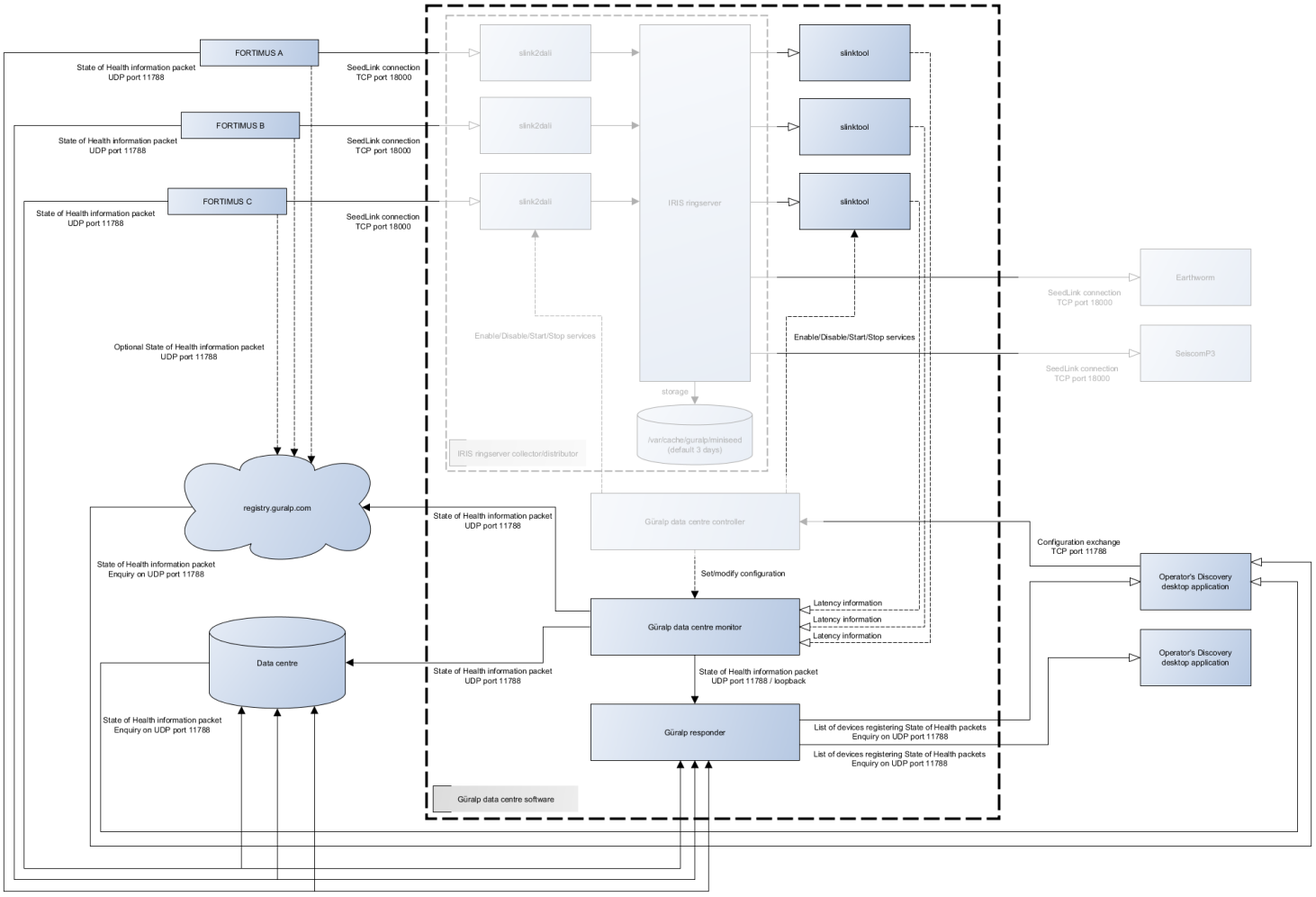
9.3.3. Configuration Management



Configuration of Data Centre software package components can be done, under normal operation, by the Discovery desktop application. Discovery provides functionality to configure which seismic stations Data Centre should be connected to and what conditions should be used to generate state of health information. Configuration exchange is performed on port 11788 TCP connection between Data Centre and Discovery. Discovery requests the current configuration from the Data Centre, modifies it if required, and sends back the updated structure.

Extra configuration may be required during installation, and the possible options are described in the above Installation section of this document.

9.3.4. State of Health



State of health information can be distributed to multiple registries by both Güralp Data Centre Monitor service and Güralp seismic station on port 11788 through UDP packets. Information gathered from the system is packetized and sent to configured Güralp Responder servers to be redistributed on request.

State of health information is requested by Discovery desktop application from the registry and displayed in the application main window table. More detailed information about system status can be obtained by accessing either device dashboard (for seismic stations) or state of health dashboard (for data centre instance). More information about how to operate Discovery application can be found in Güralp Data Centre Operator Manual.

9.4. Network Protocols and Ports

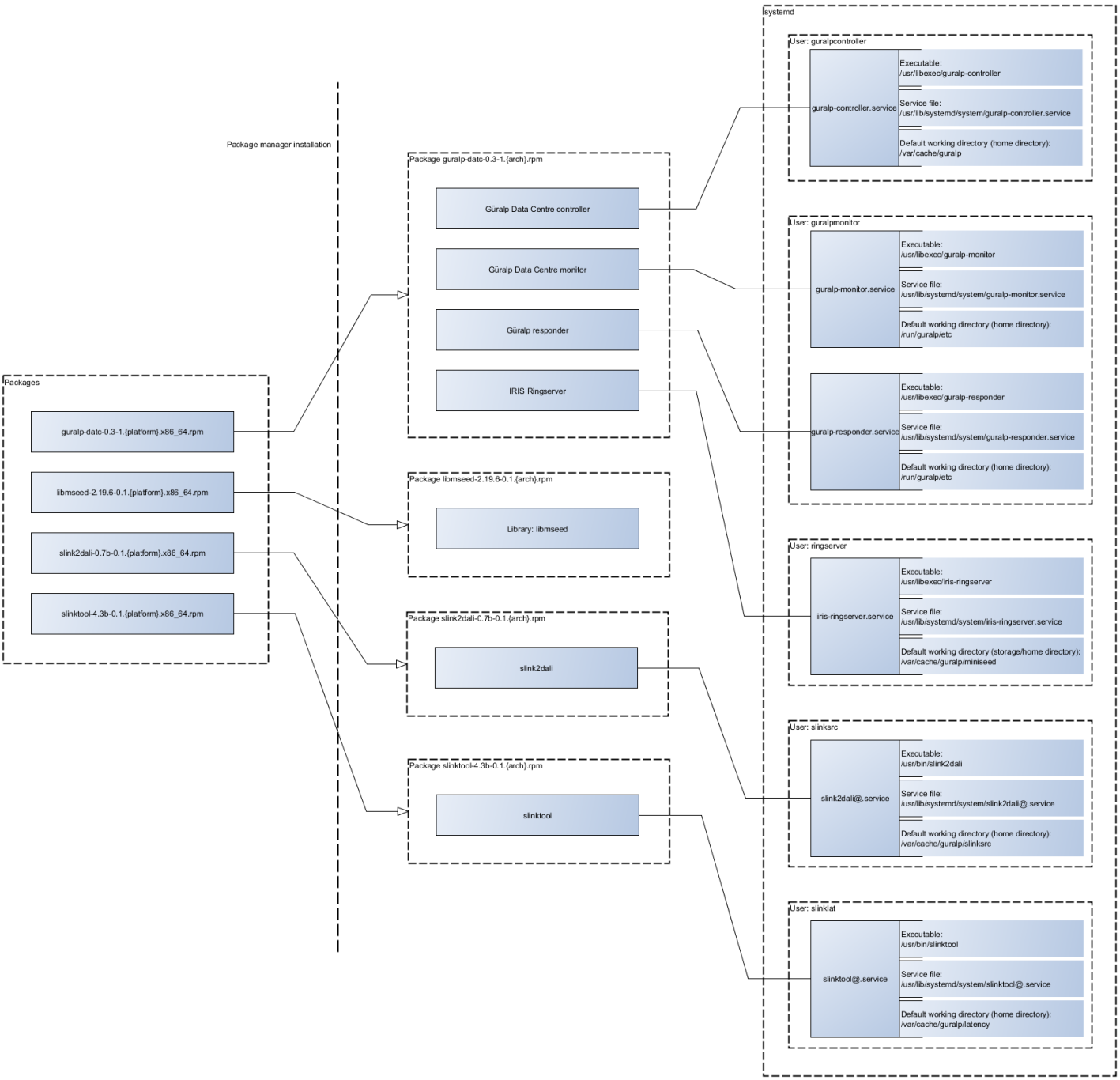
Güralp devices which are using DIG operating system require the following ports to be open/forwarded:

Port	Protocol	Description
80	TCP	HTTP server, required to access device webpage for state of health information and configuration.
1565	TCP	GDI data transmission protocol.
1567	TCP/UDP	GCF data transmission protocol.
4242	TCP	File exchange protocol used by Discovery desktop application to exchange files and configuration.
4244	TCP	Remote console used for debugging, available in Discovery desktop application.
11788	UDP	Remote procedure calls protocol used by the Discovery desktop application to remotely execute functions on the device. This port is also an outgoing port for State of Health packets that are sent to the Güralp Responder server.
18000	TCP	SeedLink data transmission protocol.

Data centre software package requires the following ports to be open:

Port	Protocol	Description
11788	UDP	Used for sending and requesting state of health information by system components and Discovery desktop application.
11788	TCP	Configuration exchange protocol used by Discovery desktop application to configure data streaming connections.
16000	TCP	DataLink data transmission protocol connection to IRIS ringserver.
18000	TCP	SeedLink data transmission protocol connection to IRIS ringserver for both incoming and outgoing data streams.
11502	TCP	Listening for State of Health packets and CAP messages by Discovery Modbus Server tool.

10. Appendix 2 – Installation Software Packages



11. Appendix 3 – Acquisition Software Components