Ref. Ares(2018)2831488 - 31/05/2018

JUMPING JIVE

Project ID: 730884

Integration existing software into central infrastructure

Deliverable:	D8.5
Lead beneficiary:	TUM
Submission date:	16/04/2018
Dissemination level:	Public

Content

1 Current project situation

The work of the TUM is split into three deliverables. The project plan defines the following timing:

M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12
^ March 2017 (Start)				^ Aug. 8th, 2017 D8.4							
M13	M14	M15	M16	M17	M18	M19	M20	M21	M22	M23	M24
^ April 15th, 2018 D8.5											

	D8.4	Evaluation of software packages
	D8.5	Integration of existing software into central infrastructure
	D8.6	Completed monitoring schedule

1.1 Project job positions and applications

The project currently suffers from the good job market conditions in Germany. A job advertisement focused on print media was not successful in the beginning of the project in March 2017. No suitable applicant could be found. Announcing the situation to students brought one applicant for a student job in the field of the project. Another job advertisement focused on online media also was not successful in March 2018.

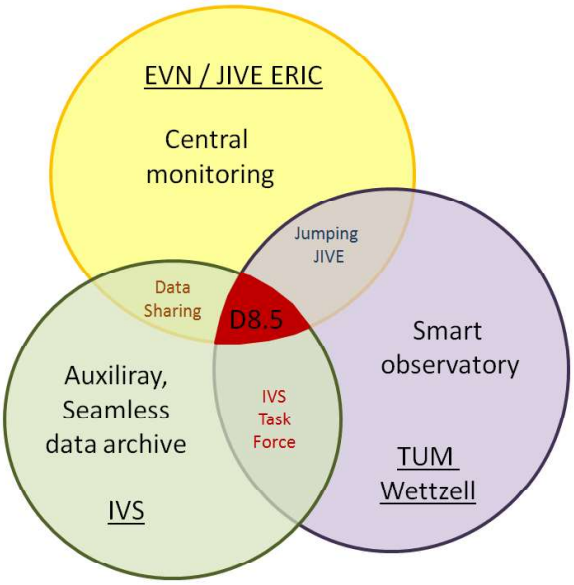
There are now attempts to get additional student workers for developments of adapters and integration preparations.

Therefore, the employment situation of people working on the project is this:

No.	Name	Position	Task	"Jumping JIVE" Allocation
1.	Edoardo Barbieri	Student worker; 8 hours per week	Development and integration tests	
2.	Dr. Alexander Neidhardt	Task responsible; Permanent position at the TUM; about 15% project work in 2017; about 90% in the first months in 2018	Project responsibility; Integration; adaption for Wettzell;	
3.	Stefanie Daurer	Secretray	Administration	
4.	Prof. Urs Hugentobler	Head of the research group	Administration; Authorized to sign	

Due to the limited manpower situation and the fact that the position of Alexander Neidhardt is mainly the development and integration of software at Wettzell, we

focused on the important integration parts with the most benefits for all cooperating partners in Jumping JIVE (but also cooperation partners of TUM Wettzell with similar requirements). This gives the following overlaps:



The parts contain:

- EVN/JIVE ERIC: Implementation of a central monitoring with real-time gathering and state evaluation for data which are relevant for operation and diagnostics
- Wettzell observatory (BKG/TUM): Implementation of a smart observatory with autonomous sessions
- IVS: Implementation of an auxiliary, seamless data archive for data sets, which are relevant for data analysis

The current deliverable implemented the basic environment and performed integration tests to prepare the establishment of a production system. It is not yet a completely expanded monitoring system and just shows the general principles. The system is going to be extended in the coming months.

1.2 Focus of the deliverable

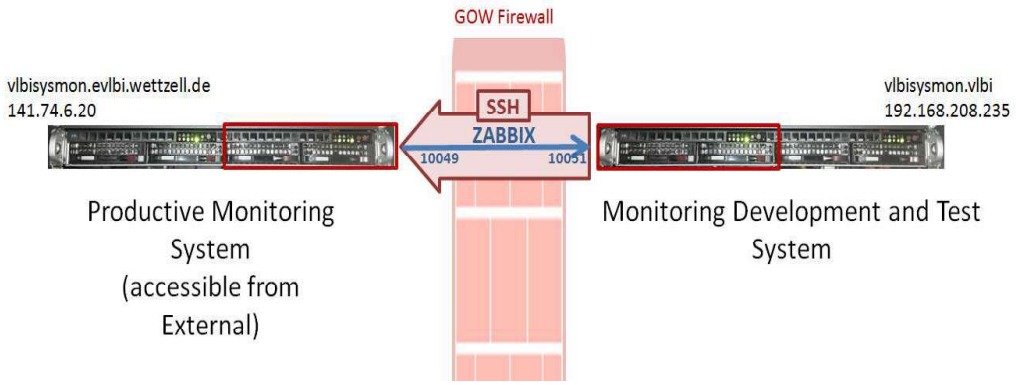
The work of for the deliverable focused on the following parts:

- Testing the integration of elements which are required for the monitoring (like maps, screens, data acquisition, etc.)
- Identifying of necessary software to integrate external systems
- Developing of necessary software to integrate external systems (this will also be continued on the following deliverable)
- Acquiring data from dummy systems and systems
- Identifying issues and limits for future use

1.3 Implementation of a test and integration infrastructure

The integration tests were done at two different locations:

- At the Wettzell observatory, where the main infrastructure is located and where one monitoring control center might be located. The infrastructure looks like this:



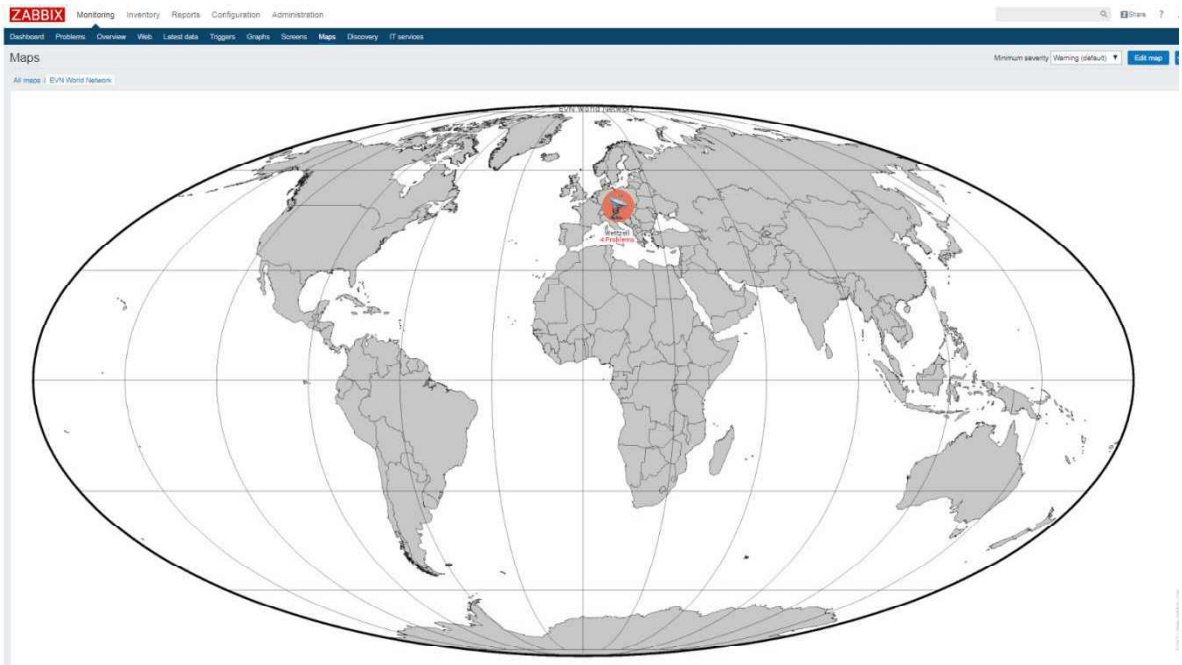
- At the TUM institute at Munich by the student worker. The infrastructure there is a desktop PC and a mobile computer with a completely installed monitoring environment consisting of ZABBIX, SysMon, MoniCA, Grafana, InfluxDB

The general principle of the centralized monitoring is a setup using different layers with different views.

The productive test environment is accessible at:
https://vlbisysmon.evlbi.wettzell.de
Username: JIVE
Password: *the password can be requested from A. Neidhardt and is sent separately in the email about the releasing of the deliverable*

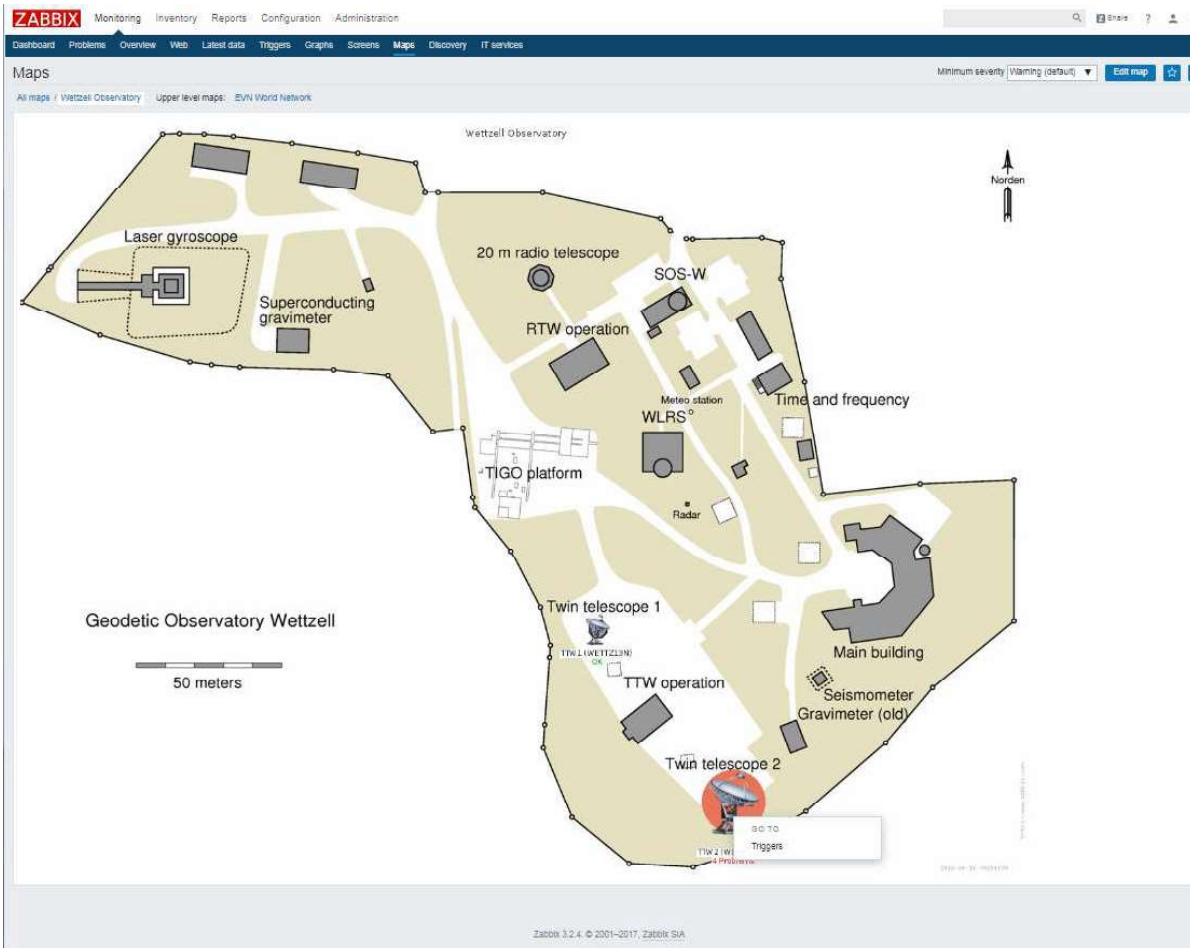
The user just has read rights. He can create new screens and maps combining existing graphs and latest data for which he got access rights. All installation work and the integration of new devices can only be done by an administrator. There is currently just “Admin” as administrator.

After login, the following screen is shown (here showing an error for the Wettzell observatory).



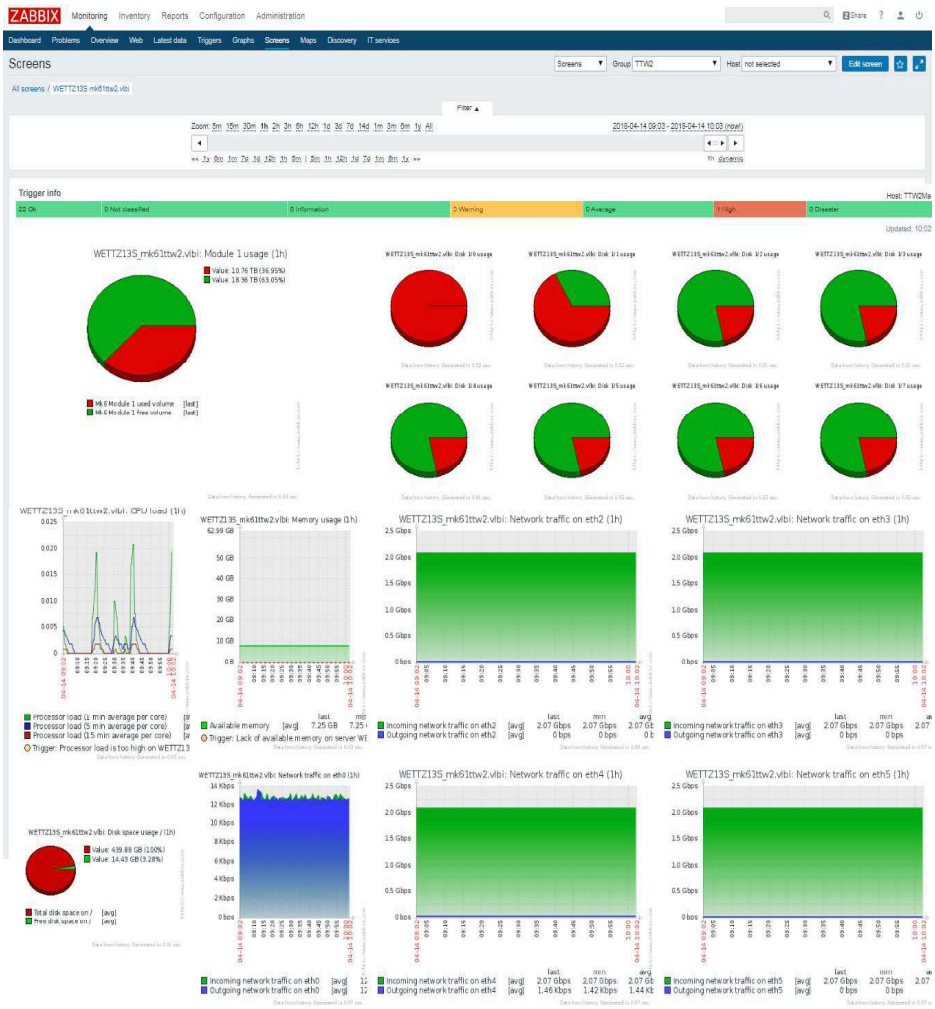
The complete monitoring interface is organized hierarchically. The main parts are “maps” with links to the following layer of maps. The link can be activated by clicking on the individual icon of an antenna or another monitoring device (“host”).

The following map can be an antenna or a whole observatory. For the Wettzell test environment, the next layer is the whole observatory.

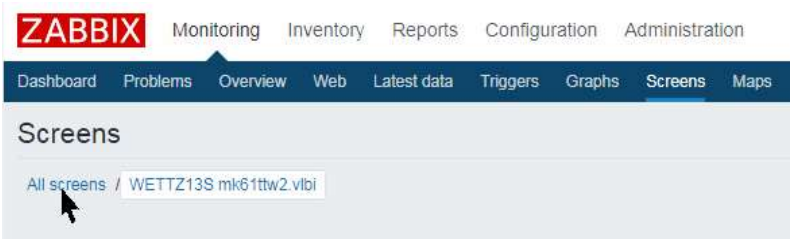


As each map contains icons to hosts or sub-maps, one can go step by step down to the system views. The complete layered setup is still under construction to be tested for the Wettzell systems. An overview of all currently installed maps can be requested by clicking to “All maps”.

In any case, the final detailed layer is a view using single “screens” for each monitoring device (“host”) with detailed plots, graphs and information, like here as excerpt for the Mark6 data recorder.



An overview of all currently installed screens can be found by clicking on “All screens”.



The number of maps and screens will be extended in the coming project periods to implement a complete working version of a system monitoring.

The following section describes all integration tests performed for deliverable 8.5.

2 Integration tests

Several integration tests were made with different test machines to find ways for getting data into the monitoring system while having different data sources at the sites. Not all elements are already installed on the productive system. But there is a detailed description in the Appendix section, how to get the systems working.

As described in deliverable 8.4, the focus was laid on system monitoring and not primarily on remote control software. As detected, the use of

- ZABBIX & SysMon
- Telegraf – InfluxDB – Grafana
- MoniCA

are most valuable.

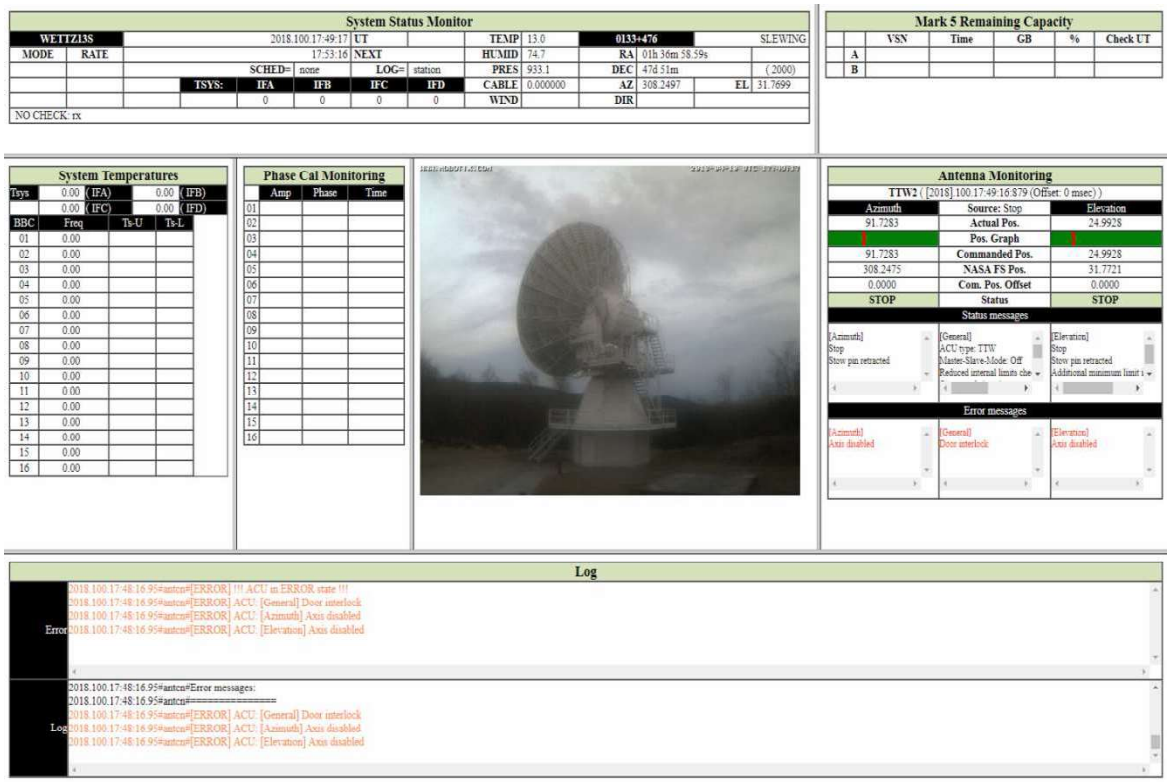
The decision was made to use ZABBIX as main monitoring system, because it has the best user comfort and a wide community within the industrial systems. ZABBIX is free of license costs.

SysMon is only used if data should be saved in files to create an archive. It is also ideal for Wettzell-like systems, to easily get data for decision processes within the systems based on remote procedure calls.

Telegraf – InfluxDB – Grafana can co-exist to ZABBIX. If GRAFANA is used as graphical user interface for all systems, ZABIX and InfluxDB, there is no additional requirement for any changes. Additional, there are ongoing tests, using "influxdb-cpp" (written by Zhang, Shanghai; see https://github.com/orca-zhang/influxdb-cpp), to directly read data from the database InfluxDB. This makes it possible to take data from there and inject it into ZABBIX and the usual interface of ZABBIX.

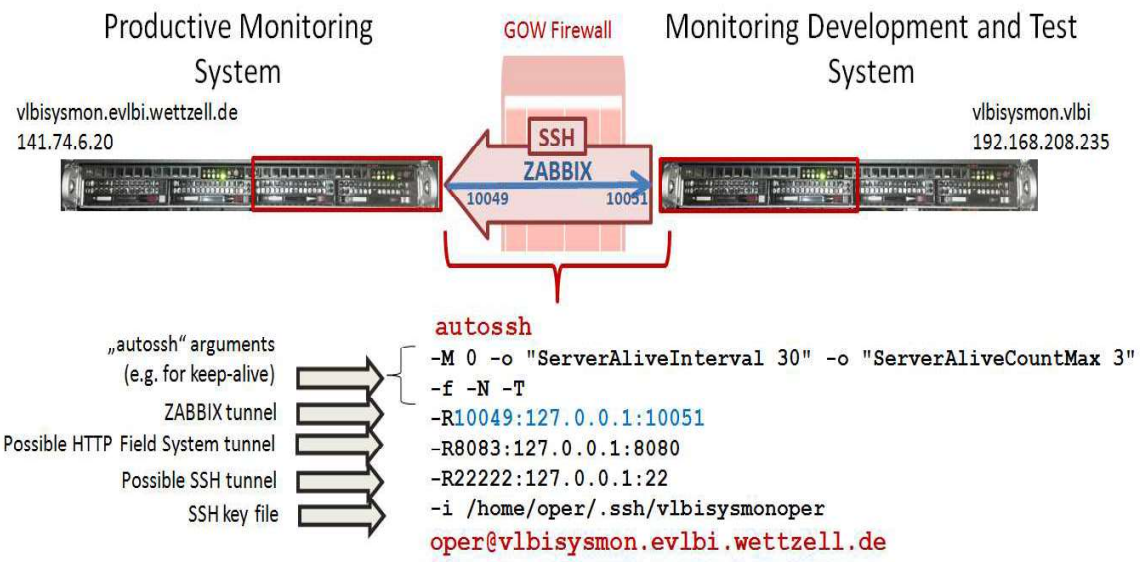
To integrate MoniCA, an adapter code was written, which currently exists in beta version and which is just tested with dummy data. It accesses the MoniCA system and can be used to convert the found device descriptions into a template configuration, which can directly be sent to ZABBIX/SysMon. There is an open task for testing with real data, using input from the Australian AuScope network.

To simplify the access to the NASA Field System without Telegraf – InfluxDB – Grafana and to support a live view on system status data, e-RemoteCtrl was extended with a web server application, so that the operational data from the NASA Field System can now be requested with browser applications. The returned web pages contain tags which can be used to filter status data. These values can then be injected to ZABBIX or ZABBIX/SysMon. A sample screen, using the Wettzell antenna WETTZ13S is shown here:



The web service might be one of the most valuable ways to get data from the sites, while potentiate also the value for the sites, having (at least read) access to their antennas via browser.

The general infrastructure consists of a centralized data collection server to which computers at the antenna sites can send data. The central server is administrated by the monitoring service. All others (like operators, etc.) just have read rights to see states or request data for diagnostics. The server offers different access methods, which can be used by the antenna sites to send data. In best case, the sites connect to the server, using secure shell (SSH) to pass firewalls with settings for tunnels through which the data flow is organized.



To keep the SSH tunnel alive and open, “autossh” can be used with parameters to periodically send keep-alive messages. The access to the central monitoring system is only possible using HTTPS or SSH with a key file. All other ports or access methods are blocked with firewall rules.

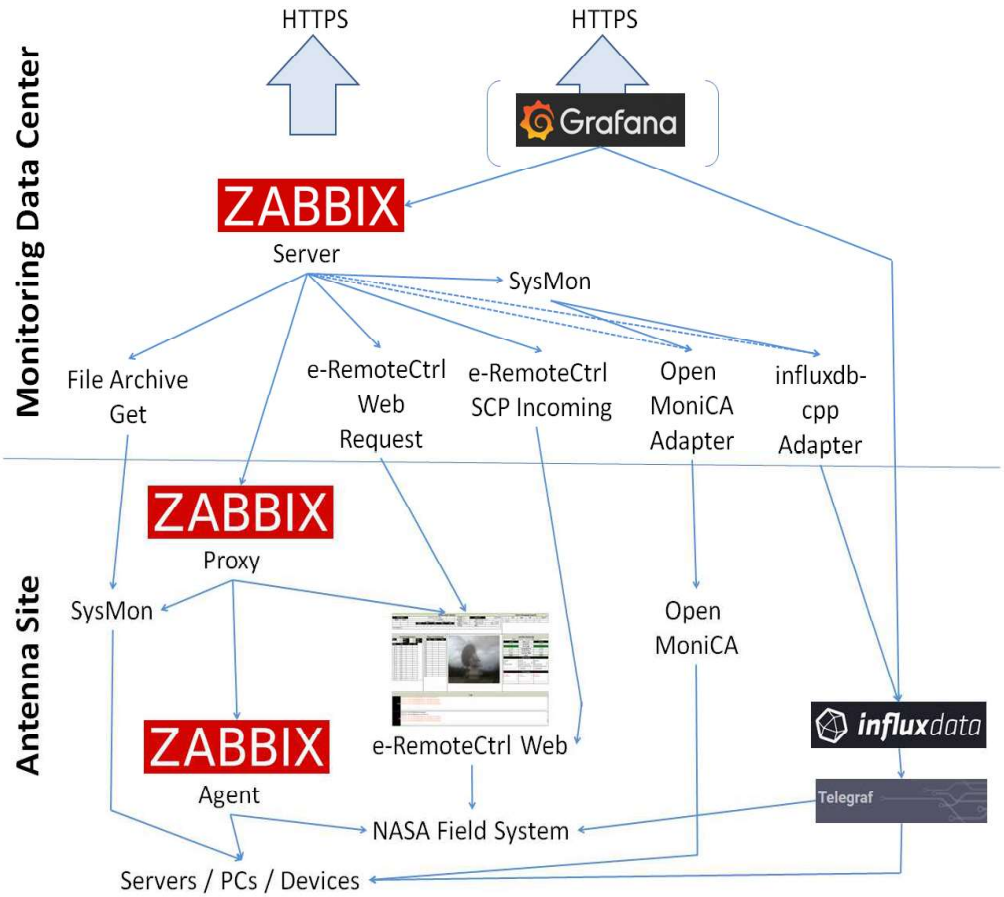
Access keys can currently be requested from Alexander Neidhardt, Wettzell observatory.

The access methods can be extended in the future. Currently, the following methods were tested for D8.5 described in detail in the following section.

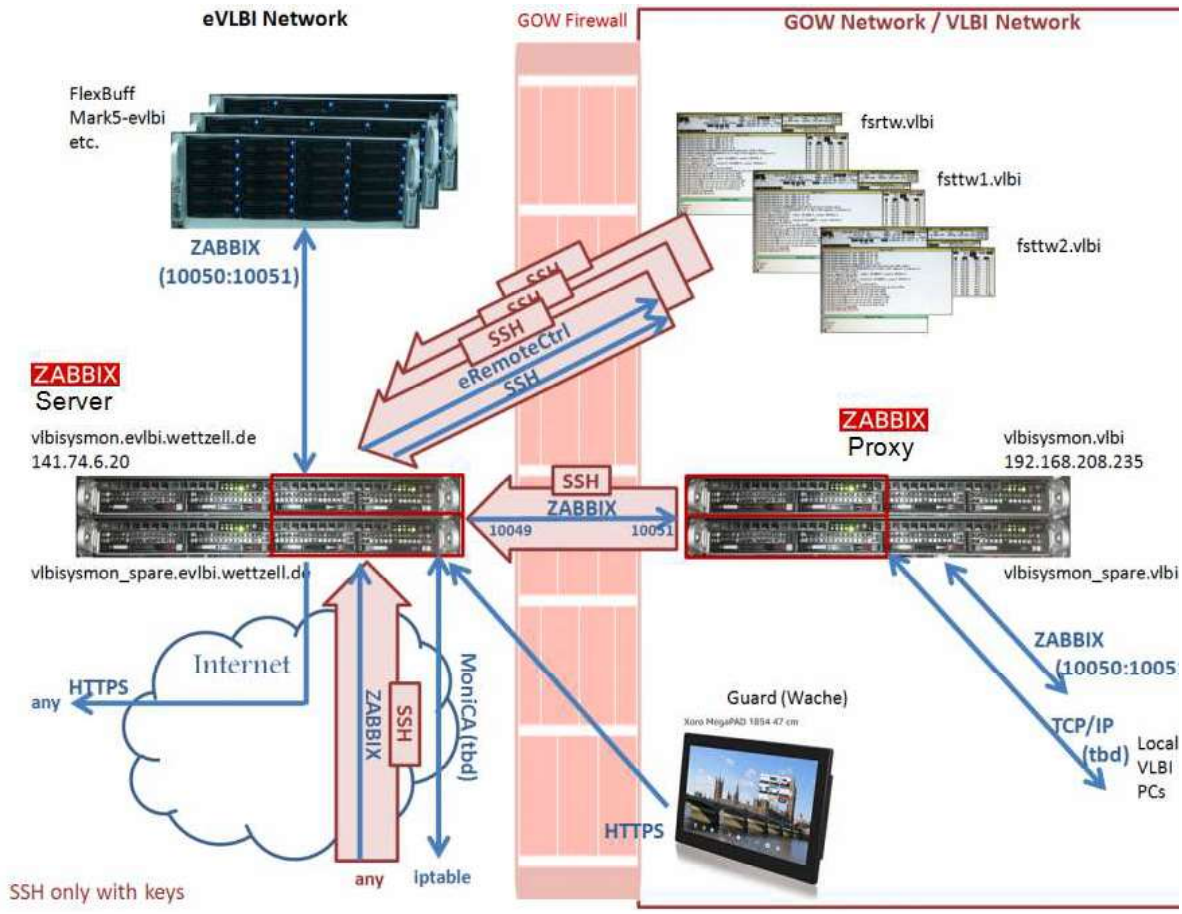
No.	Name	Test location*	Produc-tive**	Success-ful***	Future plans
1	ZABBIX: ZABBIX agent only on FS with separate ZABBIX proxy	Wettzell int. / ext.	YES: server data PARTLY: FS data	YES	Completely integrate on the productive system, because this will become the setup for the smart observatory
2	ZABBIX: ZABBIX agent only on FS with ZABBIX proxy on the FS	like 1	like 1	YES	-
3	ZABBIX: ZABBIX agent on each PC and ZABBIX connections to other devices with separate ZABBIX proxy	Wettzell int. (e.g. SNMP)	YES (inter-nally)	YES	Accessible on the external productive system
4	HTTP e-RemoteCtrl: no ZABBIX at antenna site and communication via HTTP	Wettzell int.	NO	YES	Accessible on the external productive system
5	SCP: no ZABBIX at the antenna site and communication via Secure Copy	Wettzell int. (and IVS Live)	NO	PARTLY: bug-fixing necessary	Adaption accordingly to the requirements
6	ZABBIX/SysMon: separate SysMon client for data requesting and injection	Wettzell int. / ext.	YES	YES	-
7	MoniCA: use of the adapter software to request and inject data	Munich	NO	YES	Tests with AuScope
8	Grafana: use of Grafana as additional user interface to integrate TIG and ZABBIX	Munich; Wettzell int. (SLR)	PARTLY: SLR	YES	Depending on the tests with “influxdb-cpp”
9	InfluxDB-ZABBIX: use of “influxdb-cpp” to access InfluxDB and inject data to ZABBIX	Munich	NO	PARTLY	Further tests are required

* Location of the test integration (Munich = student environment; Wettzell ext. = productive system at Wettzell; Wettzell int. = development system at Wettzell)
** The element is used for production data at Wettzell
*** Test was successful or not
**** Future plans

The test-wise integrated system combination is shown in the following image in a simplified form.



The final monitoring system will be a combination of these scenarios above. Sites should be able to have different, standard access possibilities to the system. The current environment in total is shown in the following image.

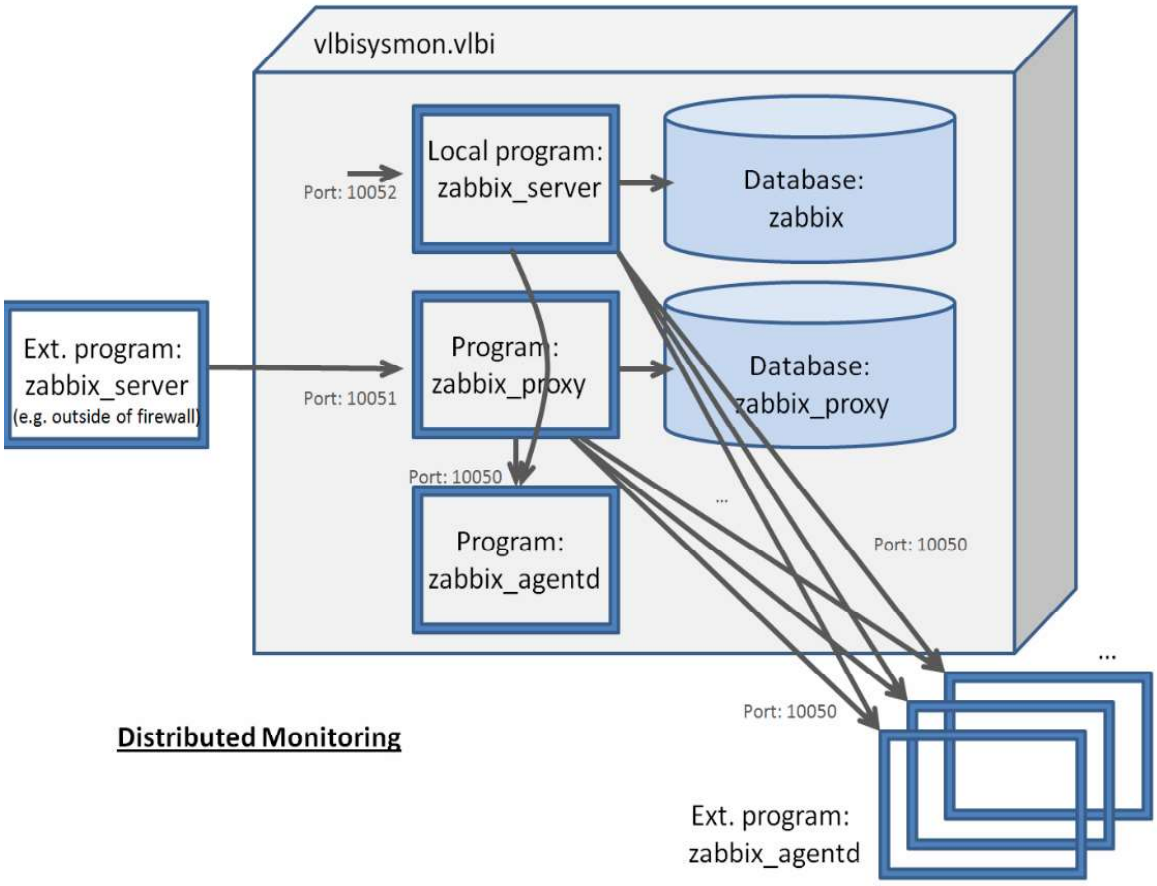


Please note for the following detailed description, that a connection for a direct communication without secure shell (SSH) is additionally possible, for specific sites. As a direct communication reduces the security of the data and access it is not separately described here, because it is not the preferred way. Each direct communication requires the setting of firewall rules at the location of the monitoring server and the antenna site.

Important is that the monitoring server just collects data, presents them and evaluates them due to predefined alert levels. It takes no control or responsibility for hazards or destructions at the location of the antenna. The server does not take over control from the site. It is just a method to reduce the reaction time in case of an issue or problem. Each antenna site should evaluate the following data classes:

Priority	Class	Description	Implementation	Example
1	Hazards to humans (Disaster)	A human being can be injured or killed	In the antenna or hardware and certified	Emergency stop, door switches
2	Hazards to systems (Disaster, High)	Hardware can be damaged	In the antenna or hardware and certified	Over-tem-peratures; grease level underflow
3	Hazards to products (High)	The creation of products is influenced or stopped	System software and hardware; NASA Field System	Servo failure, so that the antenna does not move
4	Hazards to data quality (Warning)	The quality of the data is affected	System software; additional tools	Low SNR; missing PCal; Pointing

The integration design on one monitoring machine is shown in the following image.



The ZABBIX agent (“agentd”) listen port is 10050 (standard port). Agents can be requested by several servers and proxies. Agents are in passive mode, so that the server must request data and that the agent does not send data automatically.

The ZABBIX Proxy listen port is 10051 (standard port). Proxies are always in passive mode, so that the server must request data and that the proxy does not send data automatically. One proxy can exactly be requested by one server. There is a proxy for each external data requester.

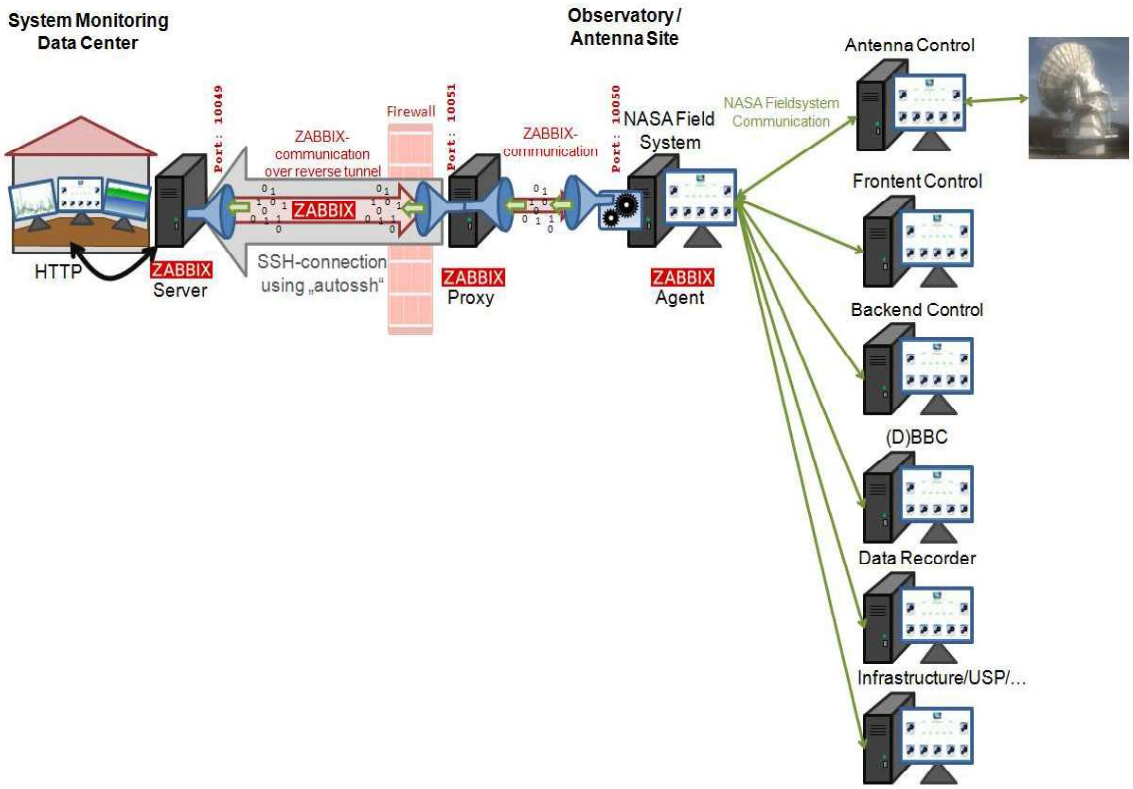
The ZABBIX Server listen port is 10052 (special for VLBI system monitoring). A server can request data from several proxies and agents.

The communication through the firewall uses a reverse SSH tunneling. It is initiated by the proxy using “autossh”, which logs in to vlbisysmon.evlbi.wettzell.de. It creates a port 10049 on vlbisysmon.evlbi.wettzell.de which is connected to port 10051 on the proxy. The ZABBIX server on vlbisysmon.evlbi.wettzell.de connects to 127.0.0.1:10049 to get data from the proxy.

2.1 ZABBIX: ZABBIX agent only on FS with separate ZABBIX proxy

No.	Name	Test location*	Produc-tive**	Success-ful***	Future plans
1	ZABBIX: ZABBIX agent only on FS with separate ZABBIX proxy	Wettzell int. / ext.	YES: server data PARTLY: FS data	YES	Completely integrate on the productive system, because this will become the setup for the smart observatory

Architecture:



Description:

The antenna site uses the NASA Field System or another equivalent system to control the observation and hardware. The communication and data acquisition between the Field System and the hardware is independent from the monitoring system. The PC with the Field System also runs a ZABBIX agent on the standard port. It monitors all PC parameters defined by ZABBIX and additional, separate operational parameters. The data additionally monitored are defined in the configuration file of the agent. Entries like

```
UserParameter=fs.meteo_humidity,/usr2/st/bin/getfsitem MeteoHumidity
```

defines an external program “getfsitem” called by ZABBIX agent each time the server request the item fs.meteo_humidity. The program can be user code written by station staff or a program provided by the monitoring center. There are several libraries to access the shared memory of the Field System, like the e-RemoteCtrl version “fsmonitor.cpp/.hpp”.

A ZABBIX proxy is installed on a separate machine at the location of the antenna. The proxy is the stub for the external ZABBIX server at the monitoring center. The proxy machine is behind the firewall of the antenna site and opens an SSH connection to the server machine. It sets a reverse tunnel from a port on the server machine to the proxy machine which is the connection over which ZABBIX communicates to request status data, set configuration parameter and reply current values. The proxy buffers the values and forwards them to the server, so that data are not lost even when the communication is shut down for a while.

The monitoring operator uses the ZABIX web frontend and the predefined screens and maps to get an overview of the current states in the whole network. The site operator can decide which data are offered to the monitoring center (by changing the definition in the configuration file) and if they are monitored (activated agent and activated tunnel).

The complete configuration of request intervals, use and management of items is done by the server and configured with the ZABBIX web frontend on the server machine.

If data should be used during a further processing, they must be requested using the ZABBIX API with JASON HTTP requests, as described in the ZABBIX documentation.

Current status:

The setup is currently implemented as productive system and test setup at Wettzell observatory. It is used for basic monitoring data. Field system data are currently not yet integrated. But the data propagation of this method is tested and evaluated using the Mark6 data recorder with similar conditions.

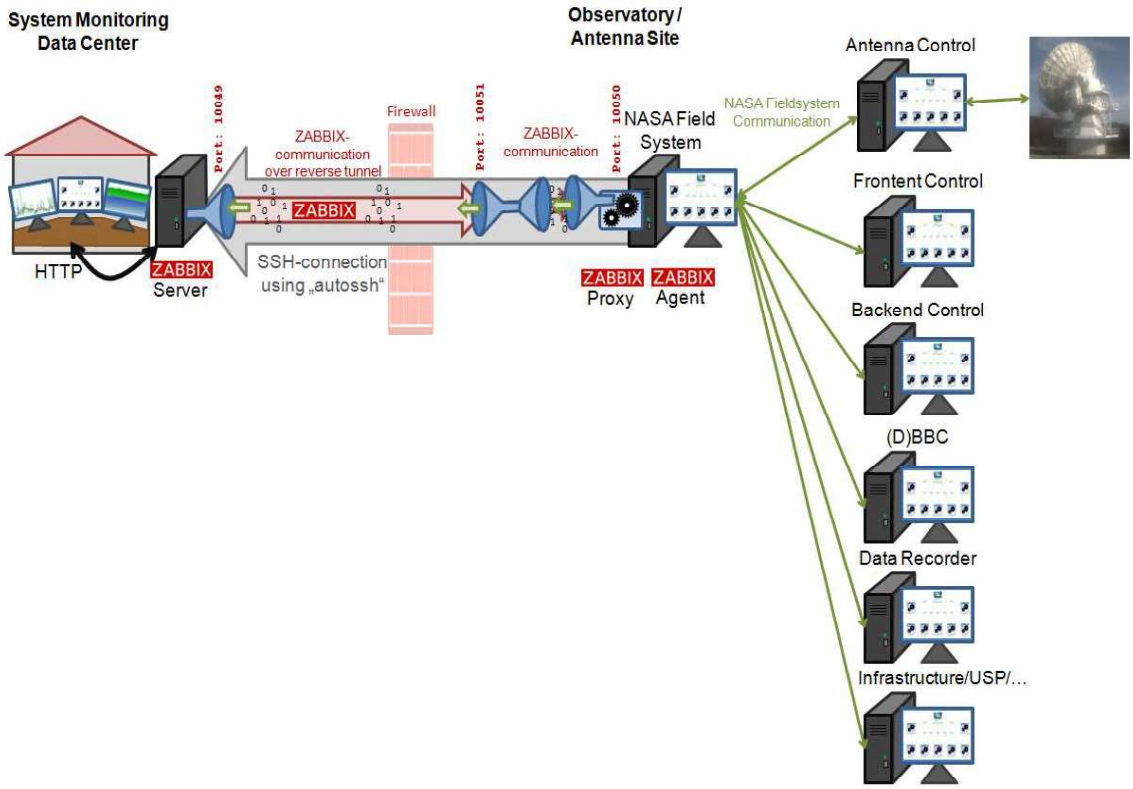
Future plans for the central monitoring:

This will be one of the main monitoring setups.

2.2 ZABBIX: ZABBIX agent only on FS with ZABBIX proxy on the FS

No.	Name	Test location*	Produc-tive**	Success-ful***	Future plans
2	ZABBIX: ZABBIX agent only on FS with ZABBIX proxy on the FS	like 1	like 1	YES	-

Architecture:



Description:

The setup is the same like in the previous section. The only difference is that the ZABBIX agent is directly on the Field System machine. This avoids an additional computer, but opens a direct connection to the Field System machine, which brings all the traffic to the main computer controlling the observation. It also requires an additional installation of the ZABBIX agent and proxy software on the Field System machine.

Current status:

The setup was experimentally tested on a separate Field System machine. It might be dependent which compiler version is used, if ZABBIX is translated from the sources on older machines in the network.

If data should be used during a further processing, they must be requested using the ZABBIX API with JASON HTTP requests, as described in the ZABBIX documentation.

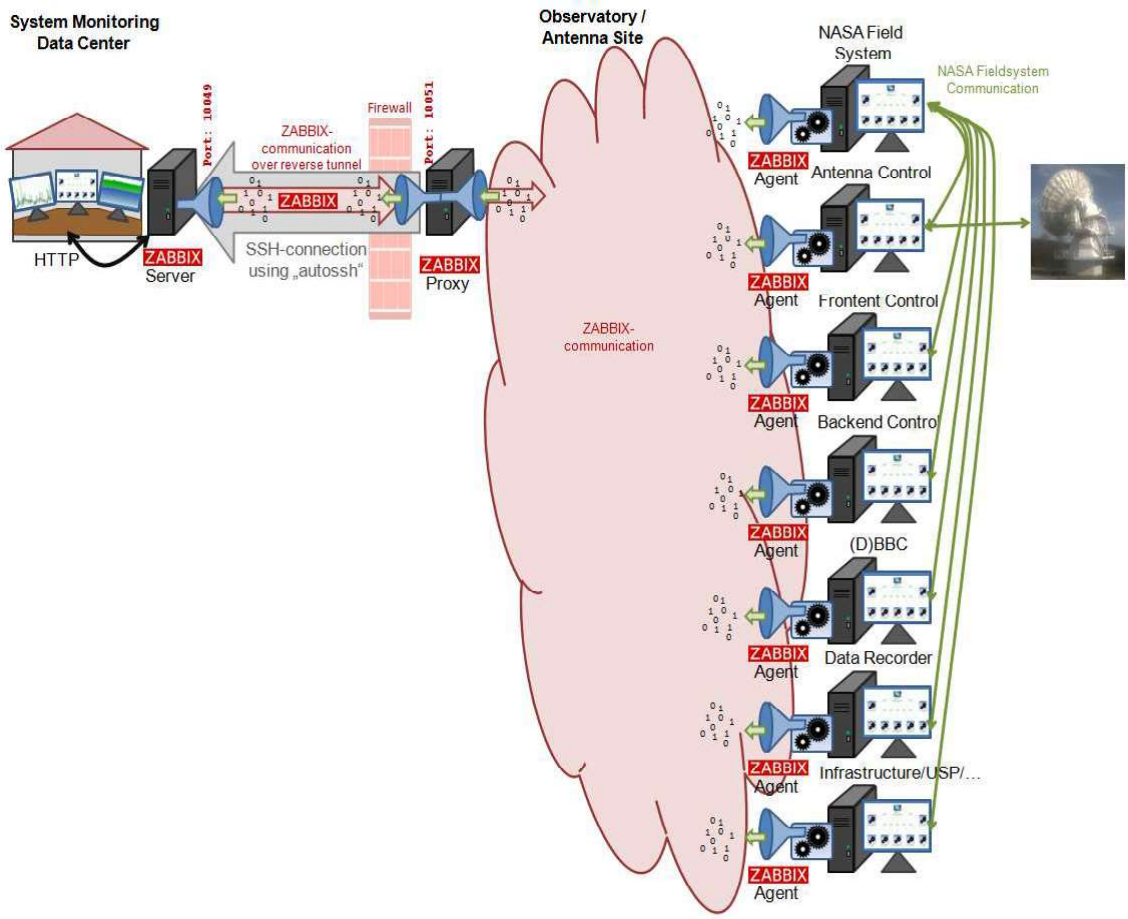
Future plans for the central monitoring:

This is one option for external sites without installing a separate, new server hardware.

2.3 ZABBIX: ZABBIX agent on each PC and ZABBIX connections to other devices with separate ZABBIX proxy

No.	Name	Test location*	Produc-tive**	Success-ful***	Future plans
3	ZABBIX: ZABBIX agent on each PC and ZABBIX connections to other devices with separate ZABBIX proxy	Wettzell int. (e.g. SNMP)	YES (inter-nally)	YES	Accessible on the external productive system

Architecture:



Description:

The setup is the similar to the first one. The difference is that each machine or device is additionally equipped with a ZABBIX agent to retrieve data in parallel to the usual control structures. Each machine defines the values which are offered by setting the individual definitions in the specific configuration files on the individual machines.

If data should be used during a further processing, they must be requested using the ZABBIX API with JASON HTTP requests, as described in the ZABBIX documentation.

Current status:

The setup is used for all external devices like USP SNMP requests, rack temperatures, etc. and will be continued at the Wettzell observatory.

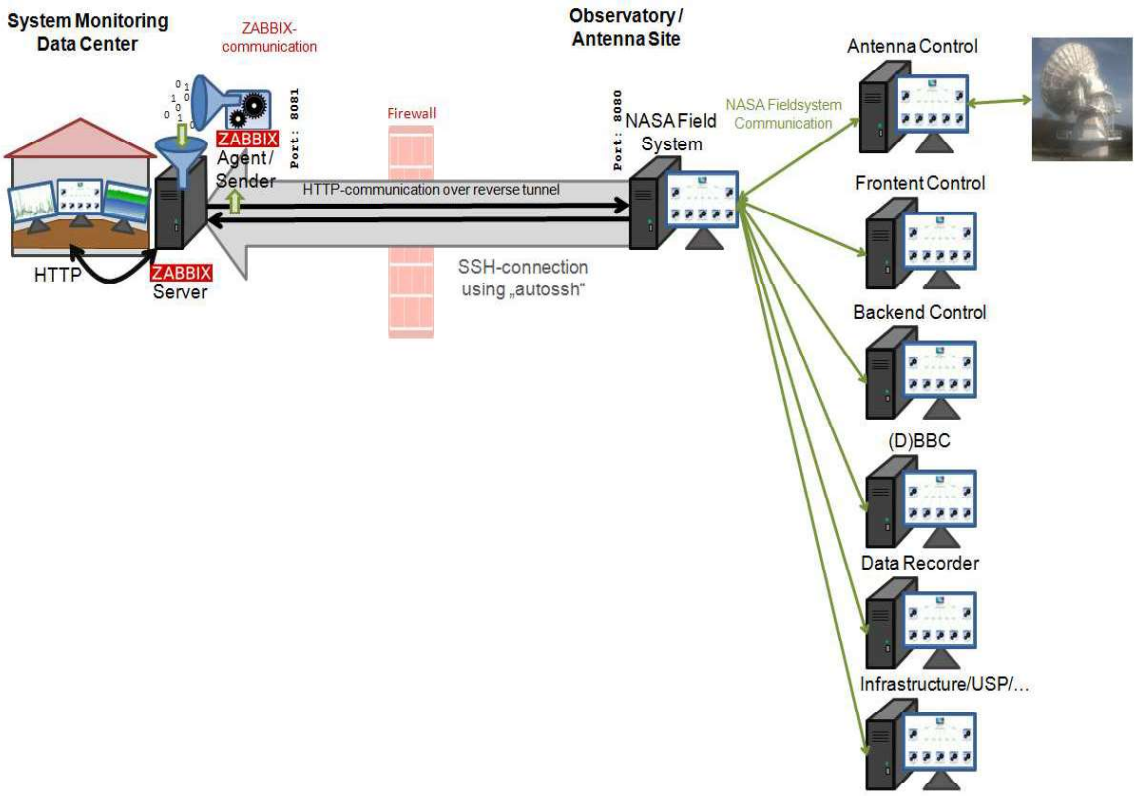
Future plans for the central monitoring:

This option will be continued at Wettzell but might be too time-consuming for other sites.

2.4 HTTP e-RemoteCtrl: no ZABBIX at antenna site and communication via HTTP

No.	Name	Test location*	Productive**	Successful***	Future plans
4	HTTP e-RemoteCtrl: no ZABBIX at antenna site and communication via HTTP	Wettzell int.	NO	YES	Accessible on the external productive system

Architecture:



Description:

The setup is similar to the second one with a major difference that ZABBIX is not required on location of the antenna. Instead of ZABBIX, the new version of e-RemoteCtrl with an integrated web server must be used. The web server uses the environment of the Field System and the implementations of the Wettzell e-RemoteCtrl software to read HTML files with comment tags which are dynamically replaced by real data.

Currently the following HTML pages are supported:

- FieldSystemMonitoring.html: the main web page (similar to index.html or the root page)

- SystemStatusMonitor_iframe.html: a page with an iframe to improve the performance of the system status web page for page updates on specific browsers like Chrome
- SystemStatusMonitoring.html: the system status web page with information like in the System Status Monitoring window on the Field System screen
- Mark5RemainingCapacity_iframe.html: a page with an iframe to improve the performance of the Mark5 capacity web page for page updates on specific browsers like Chrome
- Mark5RemainingCapacity.html: the Mark5 Remaining Capacity web page with information like in the Mark5 Remaining Capacity window on the Field System screen
- SystemTemperatures_iframe.html: a page with an iframe to improve the performance of the System Temperatures web page for page updates on specific browsers like Chrome
- SystemTemperatures.html: the System Temperatures web page with information like in the System Temperatures window on the Field System screen
- PhaseCalMonitoring_iframe.html: a page with an iframe to improve the performance of the Phase Calibration web page for page updates on specific browsers like Chrome
- PhaseCalMonitoring.html: the Phase Calibration Monitoring web page with information like in the Phase Calibration Monitoring window on the Field System screen
- WebCam_iframe.html: a page with an iframe to improve the performance of the webcam web page for page updates on specific browsers like Chrome
- WebCam.html: the webcam web page which uses a periodically fetched webcam image of the antenna
- Antenna_iframe.html: a page with an iframe to improve the performance of the antenna monitoring page for page updates on specific browsers like Chrome
- Antenna.html: the antenna monitoring web page with a standardized e-RemoteCtrl view of the antenna control unit, where a specific station code in the e-RemoteCtrl software must be written by antenna staff
- Log_iframe.html: a page with an iframe to improve the performance of the log file page for page updates on specific browsers like Chrome
- Log.html: the Log File web page with information from the session log like in the Logging window on the Field System screen plus a separated filtered error log

The web pages can be changed individually at each site. Additional links to other web servers can be integrated to extend the possibilities.

The web pages contain standardized comment tags, which are replaced by real values, e.g.

```
"<!--ANTENNA-->"
```

is replaced by

```
"<!--ANTENNA--> WETTZELL <!-- -->"
```

for the 20m Wettzell antenna.

The Field System machine is behind the firewall of the antenna site and opens an SSH connection to the ZABBIX server machine. It sets a reverse tunnel from a port on the server machine to the web service on the Field System machine which is the connection over which the ZABBIX server machine requests the web pages.

The ZABBIX server uses "external checks" to call a script or program which downloads the web pages periodically and also checks the performance parameters of the download. The script searches for the patterns in the web pages and feeds the items found to the ZABBIX server.

<u>Current status:</u>

This setup will be used for Wettzell antennas. It might be the optimal way for all other sites, because they just have to install the e-RemoteCtrl server, actvate the integrated web server and open the tunnel to the ZABBIX machine using autossh. It is currently not completely activated on the productive system.

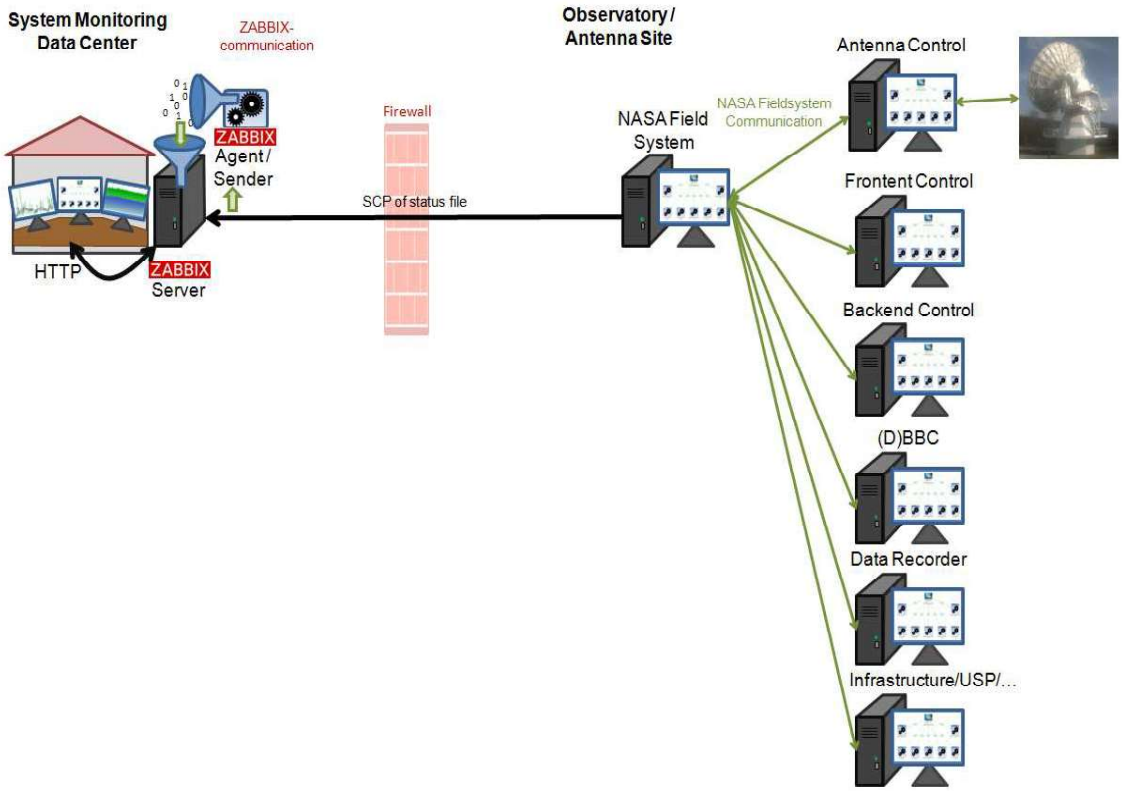
<u>Future plans for the central monitoring:</u>

The complete implementation will follow until end of April 2018.

2.5 SCP: no ZABBIX at the antenna site and communication via Secure Copy

No.	Name	Test location*	Productive**	Successful***	Future plans
5	SCP: no ZABBIX at the antenna site and communication via Secure Copy	Wettzell int. (and IVS Live)	NO	PARTLY: bug-fixing necessary	Adaption accordingly to the requirements

Architecture:



Description:

The setup uses the same idea like the previous one to avoid ZABBIX installations on the antenna site. It even avoids the creation of a permanent tunnel to the monitoring server machine. Instead it uses secure copy (SCP) to send a standardized file with all relevant values to an incoming folder on the server machine. This sending process can

- be written in any programming language by the antenna staff or
- be activated in the e-RemoteCtrl software (where only updates are sent, if there are relevant changes)

The file format is this:

```
<eQuickStatusInfo>
    Service = IVS
    Stationname = WETTZELL
    StationIVSCode = Wz
    Schedule = k14242wz
    Status = [eRC] Recording<br>
    DateTime = 2014.242.07:42:52
    TimeNext = 07:42:53
    Source = 0016+731
    Scan = k14242_wz_242-0742
    Mark5VSN = BKG-E002
    Mark5Volume = 1470.0
    Mark5Used = 0.8
    RightAscension = 00h19m45.79s
    Declination = 73d27m30.00s
    Azimuth = 337.5532
    Elevation =  43.4183
    CableDelay = 0.006511
    SystemTemperatureIFA = 34
    SystemTemperatureIFB = 98
    SystemTemperatureIFC = 32
    SystemTemperatureIFD = 0
    MeteorologyTemperature = 16.8
    MeteorologyHumidity =  86.3
    MeteorologyPressure =  948.7
</eQuickStatusInfo>
```

A program or script periodically reads this file on the server machine and sends the data to the ZABBIX server.

Current status:

This setup was tested for IVS Live. It can be used if the remote control access to e-RemoteCtrl is deactivated, because there is an access right bug which is not yet fixed. The software is not installed at the moment, due to development constraints.

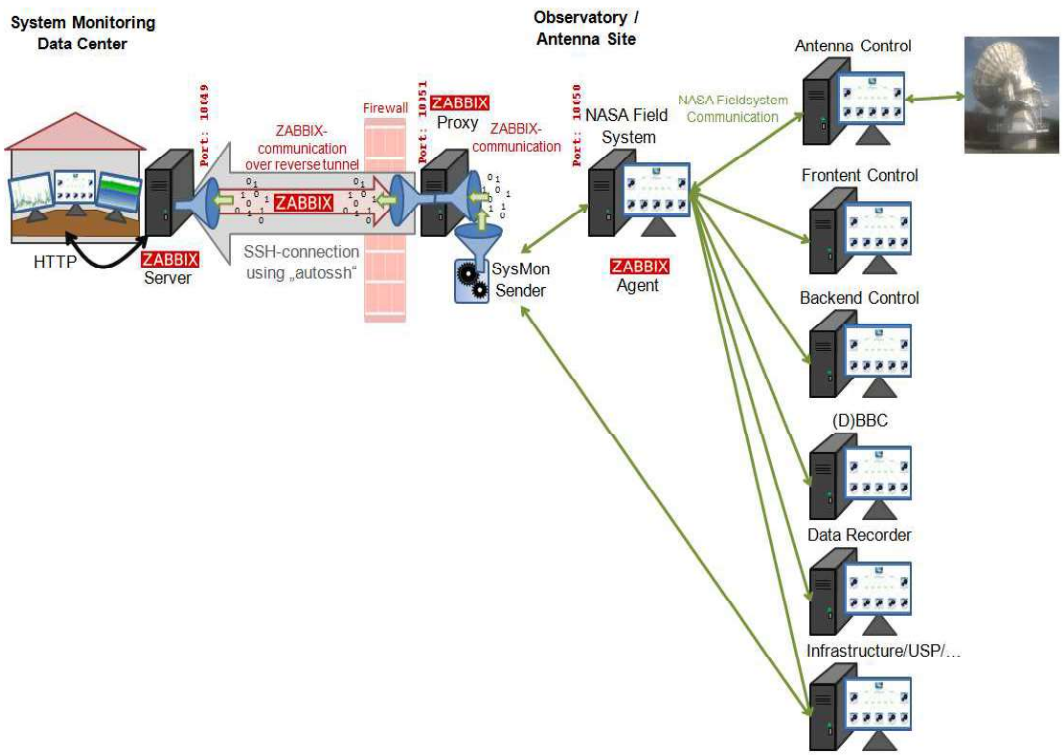
Future plans for the central monitoring:

This participation possibility might be a solution for older antennas, because nothing is required besides a small script from station staff periodically sending the information file. Until end of 2018, the software should be reactivated.

2.6 ZABBIX/SysMon: separate SysMon client for data requesting and injection

No.	Name	Test location*	Productive**	Successful***	Future plans
6	ZABBIX/SysMon: separate SysMon client for data requesting and injection	Wettzell int. / ext.	YES	YES	-

Architecture:



Description:

The setup is used for meteorological values from the antenna WETTZ13S, were data should also be archived in files. The architecture of SysMon uses a SysMon sensor proxy for this situation. This proxy is a client program which uses the proprietary communication used for the local control of the antennas to request predefined data. The request is hard-coded. Received data are taken to feed the application interface (SysMon API) of Wettzell SysMon. It is a separate database with own tables and priority classes. The application interface registers sensors, creates templates for the registration of the sensors in ZABBIX and sends data to both databases. In case of ZABBIX, it uses the zabbix_sender program. It works like an adapter between SysMon and ZABBIX.

The first step in the client is a registration phase which must be done once. It reads a configuration file and uses “usRegisterSensors” of the SysMon API to

read a configuration file and install the SysMon tables and prepare the ZABBIX template. Most important parts in the configuration file are the sensor definitions which are directly converted into ZABBIX items. Set limits are directly converted into triggers which rise alert levels. An excerpt of such a configuration file looks like this:

```
<MCISensor>
        SensorID      = WETTZ13SMeteo_MeteoboxInternalTemperature
        SensorName    = MeteoboxInternalTemperature
        SensorUnit    = deg C
</MCISensor>
<MCISensor>
        SensorID                  = WETTZ13SMeteo_MeteoboxWindSpeed
        SensorName                = MeteoboxWindSpeed
        SensorUnit                = km/h
        SensorMaxWarningLimit     = 60
        SensorMaxAlertLimit       = 70
</MCISensor>
```

The rest of the client is a low-level while loop doing the following steps:

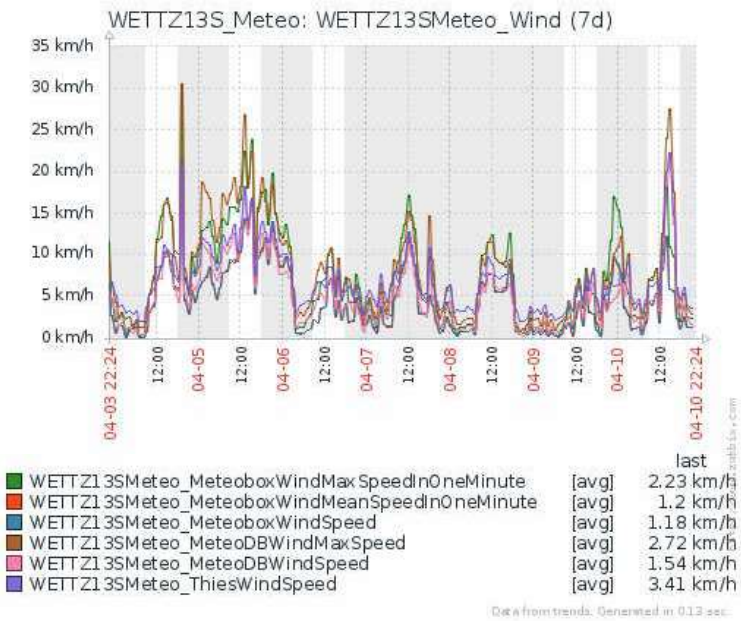
- Open connection to the data source using remote procedure calls, simple sockets, or serial connections
- Reads values
- Convert values to suitable values for SysMon API (e.g. adapt the precision)
- Use method “usSendSingleData” of the SysMon API to inject the data to SysMon and ZABBIX
- Close connection
- Sleep a specific time period (usually a second for wind data and 1 to 5 minutes for other meteo values)

The client is started with a start script and does the data processing periodically.

In case of the Wettzells productive system, SysMon sends data to the ZABBIX proxy. The SysMon sensor nodes are registered at the ZABBIX server using the generated template. Therefore, ZABBIX proxy collects all SysMon data locally, while ZABBIX server requests these data sets from the proxy.

<u>Current status:</u>

This setup is used for meteorological data. The data can be requested via the external web page, e.g. for the meteo data of antenna WETTZ13S. Wind data for one week taken from all wind sensors of the observatory look like this:



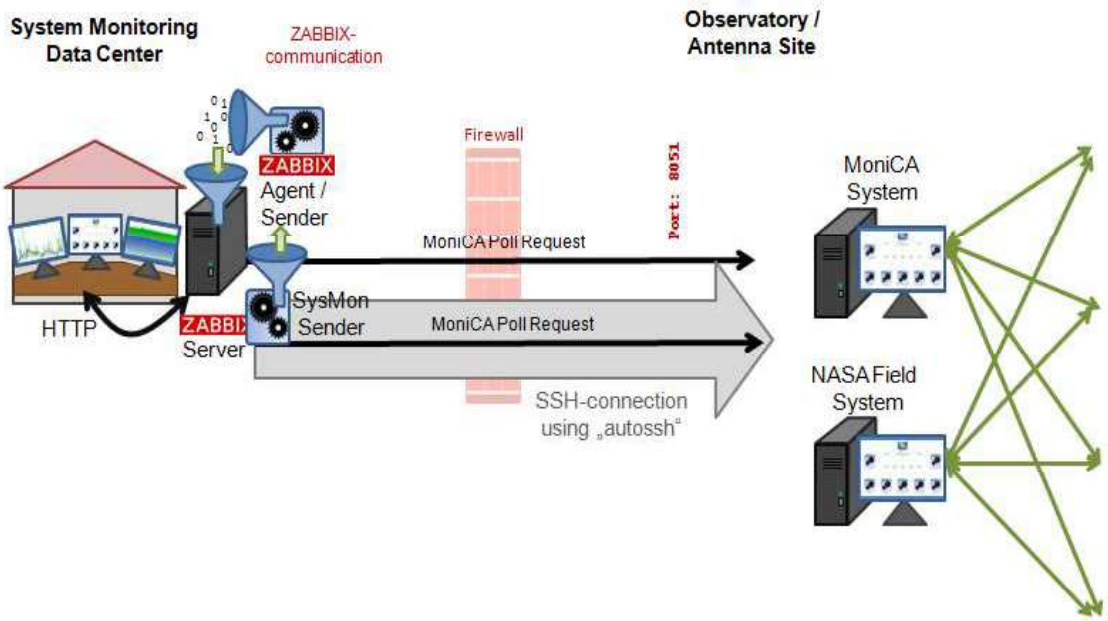
<u>Future plans for the central monitoring:</u>

The system will be implemented for all antennas at Wettzell. It might be a good solution for analysis data, which should be stored in files, like required for the IVS seamless auxiliary data archive.

2.7 MoniCA: use of the adapter software to request and inject data

No.	Name	Test location*	Produc-tive**	Success-ful***	Future plans
7	MoniCA: use of the adapter software to request and inject data	Munich	NO	YES	Tests with AuScope

Architecture:



Description:

The setup adapts to antenna systems which already use MoniCA for their own monitoring. Currently the antennas are mainly located in Australia. The application on site of the antennas is Java-based. The antenna sites do not have to install additional software. The idea for the centralized monitoring is to use a small adapter program on the server machine which polls available monitoring points and the data from the already existing systems. The program is located on the ZABBIX server machine and connects external machines at the location of the antenna to the centralized monitoring.

Currently just dummy data were used for the integration tests. During D8.5, a basic adapter class was written in C++ as demonstrator for the possibility to directly use data from MoniCA. This class must be beautified and standardized for the final implementation on a production system. Nevertheless, it shows that data can be retrieved from a MoniCA system and inserted into SysMon/ZABBIX.

Current status:

The setup is just integrated on the test system at Munich to do development purposes. It is not yet prepared for real-world data from the Australian telescopes.

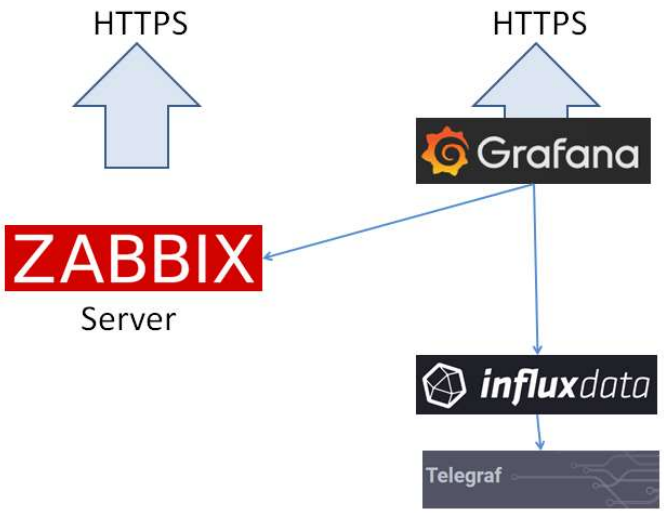
Future plans for the central monitoring:

After cleaning up the class, it should be offered as access point to MoniCA.

2.8 Grafana: use of Grafana as additional user interface to integrate TIG and ZABBIX

No.	Name	Test location*	Productive**	Successful***	Future plans
8	Grafana: use of Grafana as additional user interface to integrate TIG and ZABBIX	Munich; Wettzell int. (SLR)	PARTLY: SLR	YES	Depending on the tests with “influxdb-cpp”

Architecture:



Description:

The setup is more or less a work-around to directly support the NASA Telegraf – InfluxDB – Grafana system. It just adds Grafana as an additional graphical user interface on top of ZABBIX web interface.

Grafana offers some advantages, e.g with the extended possibility of many more graph types than ZABBIX or in case of bool states, where the steps are shown correctly. The disadvantage is the complex use which is not always intuitive.

In case of the Wettzell laser ranging systems, Grafana on top of ZABBIX was successfully tested and is in use for the internal monitoring system of the satellite laser ranging. Other tests were made in Munich on the test systems using dummy data.

Current status:

The setup is still in use at the Wettzell laser ranging system. It is not yet installed on the productive system of VLBI.

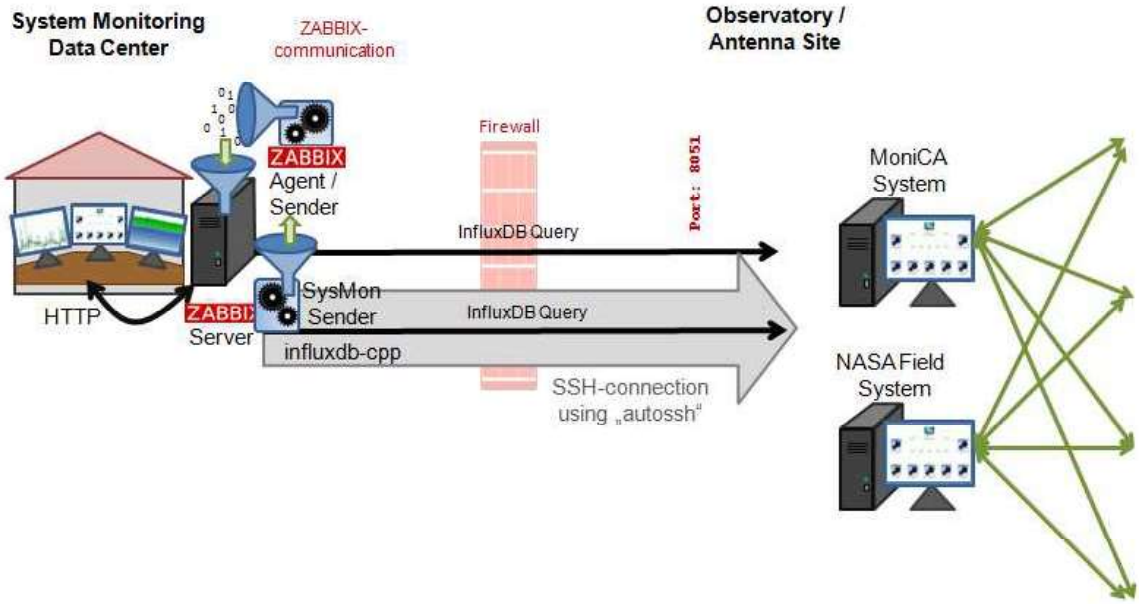
Future plans for the central monitoring:

Depending on the requirements (e.g. if the e-RemoteCtrl web server access does not solve the problem for all antennas), Grafana can be installed in addition to ZABBIX. It might also be interesting for better graphics or in combination with ZABBIX web pages.

2.9 InfluxDB-ZABBIX: use of “influxdb-cpp” to access InfluxDB and inject data to ZABBIX

No.	Name	Test location*	Produc-tive**	Success-ful***	Future plans
9	InfluxDB-ZABBIX: use of “influxdb-cpp” to access InfluxDB and inject data to ZABBIX	Munich	NO	PARTLY	Further tests are required

Architecture:



The architecture is the same like used for the SysMon/ZABBIX client with the MoniCA adapter.

Description:

The setup is exactly the same like for the MoniCA adaption. The development is still ongoing and uses an external library “influxdb-cpp”. The software architecture is identical to the MoniCA adaption.

Current status:

First tests.

Future plans for the central monitoring:

It might be a replacement for the work-around using Grafana as additional user interface, because data can be injected to ZABBIX using the adapter.

3 Conclusion and outlook

Several integration tests were successfully made to prepare a first working version of a productive system. Test systems were installed on a development PC in Munich and as system in use at the Wettzell observatory. Dummy data and real data were successfully used to test the access methods to get data into the monitoring system. The real data are taken from the Wettzell antennas, mainly the newer 13.2m VGOS antenna WETTZ13S. The productive system is accessible from the public Internet.

Due to limitations of staff, the integration mainly made a focus on integration tests and detailed descriptions, so that issues and problems could be find out and that one can use the documents to implement an own monitoring center, e.g. at JIVE. At least 9 integration tests with several sub-aspects were performed. It was necessary to write small code adapters or to combine existing software during the installation. Most of the tests were completely successful. Only a few require additional work to get them running. Almost have of the integrated methods are already usable on the productive system at Wettzell. The rest is in preparation.

It became clear during the integration, that the most valuable way with the least changes and work at the antenna sites is the use of e-RemoteCtrl web service. Therefore, main focus will be laid on such method for the next deliverable. The other methods will be integrated completely, but only activated if there are constraints which require the use these access methods.

Within the coming months, the monitoring will be extended to a complete system for the Wettzell observatory and for all three antennas. This is a first complete test-bed for a centralized monitoring. There are also contacts to test external antennas.

Finally, the test integration phase is successfully finished. Nevertheless, the integration has some overlaps (bug-fixing, integration on a productive system, etc.) with the following deliverable.

4 Appendices

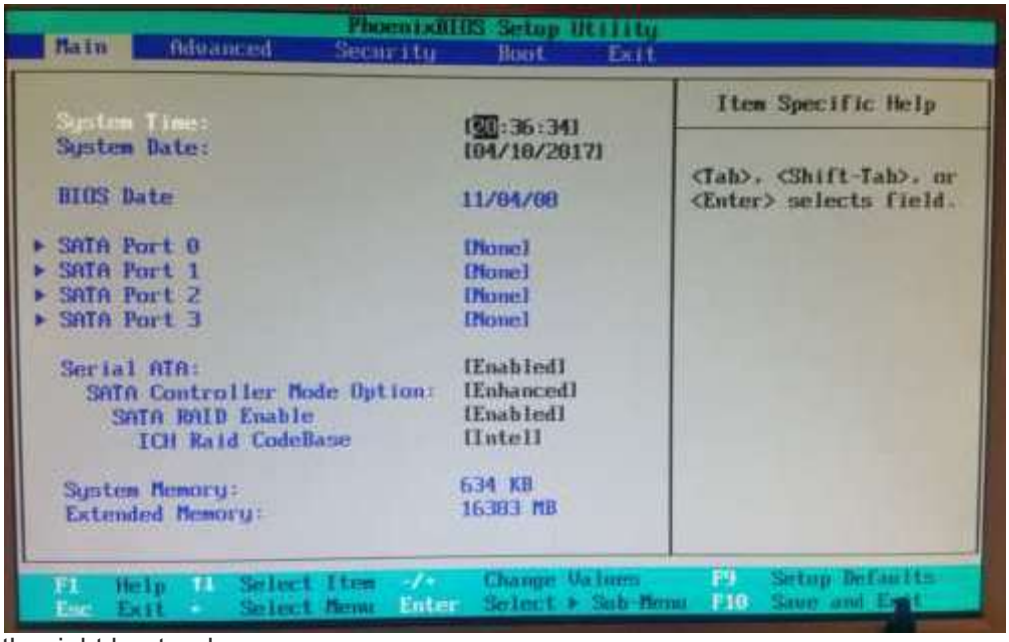
4.1 Appendix: Installation of a VLBI SysMon Node (with updates to D8.4)

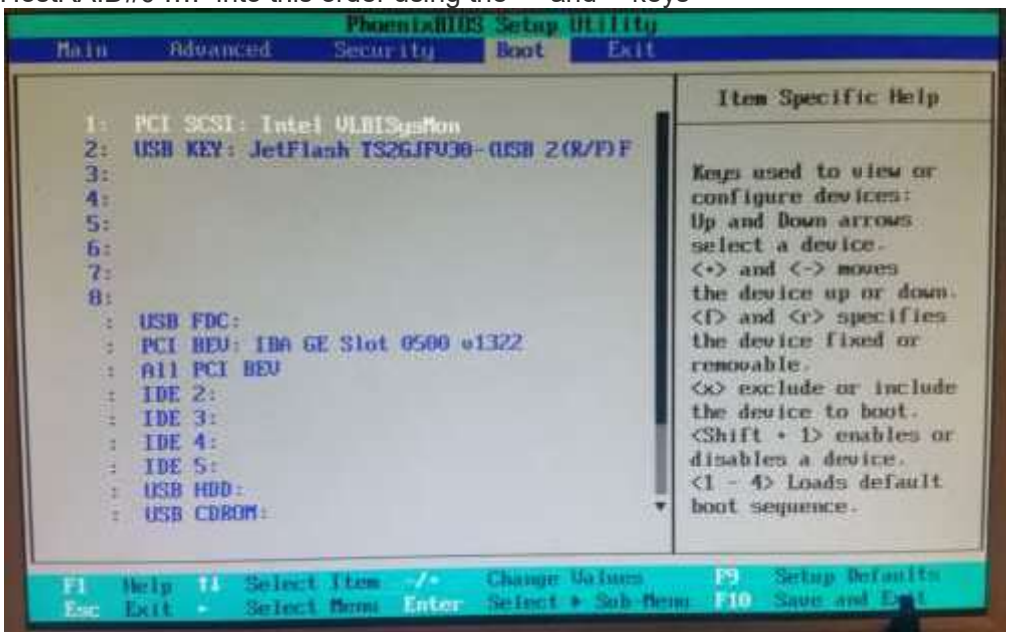
4.1.1 Used machines

- The used machines are: Supermicro X7DWT/X7DWT-INF/X7DWT-INF+ with two boards in one one-height slot
- User Manual
- 
- The completely installed SysMon server looks like this:
- 
- Both computers must be configured in the same way
  - The first is for the internal management (local telescopes)
  - The second is for the external management (other telescopes etc.)

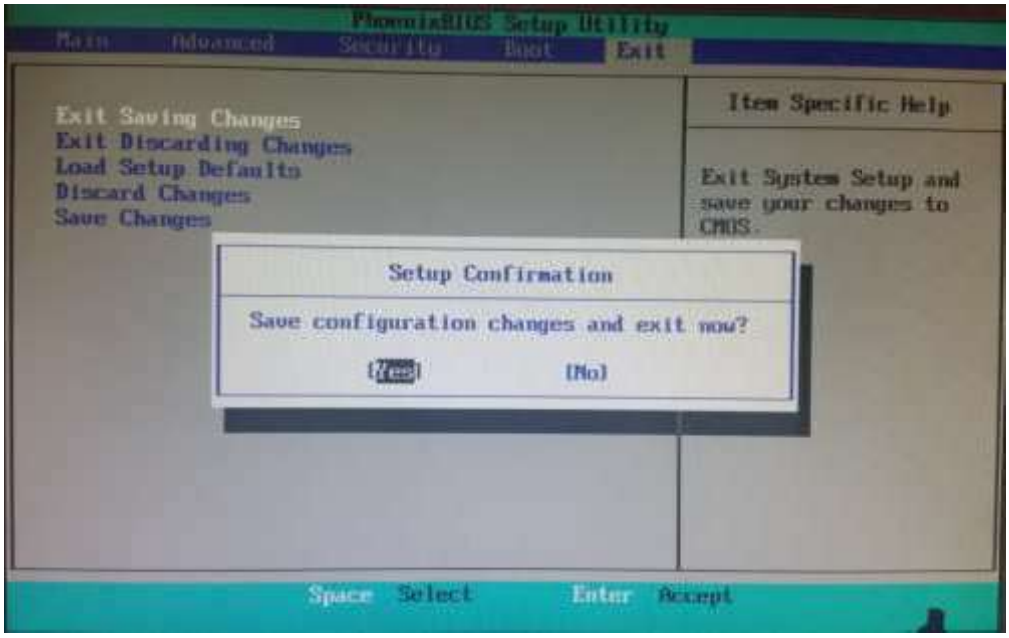
4.1.2 Setup the BIOS

- Open BIOS by pressing “DEL” after startup of the computer
- **Attention: an English keyboard style is used for the following configuration!!!**
- Activate the RAID system in the BIOS
  - See User manual
  - Enable SATA in the “Main” screen of the BIOS system:
    - “Serial ATA: Enabled”
    - “SATA Controller Mode Option: Enhanced”
    - “SATA RAID Enable: Enabled”
    - “ICH Raid CodeBase: Intel” (we use an Intel ESB2 RAID controller)

- 

- Set the right boot order
  - Change into “Boot” screen of the BIOS system using the arrow keys
  - Push “USB KEY” (maybe also “USB FDC” or other USB devices) and “ PCI SCSI: HostRAID#0 ….” into this order using the '+' and '-' keys
  - 

- Exit and save the configuration with all changes using the “Exit” screen

- 

4.1.3 Configure the RAID 0

- We use two HDDs with 1TB each as RAID 1 (mirror set) for redundancy on anIntel ESB2 RAID controller.
- **Attention: an English keyboard style is used for the following configuration!!!**
- Open the Intel RAID Configuration Utility using Ctrl+'I'
- Activate the creation of a raid
- 

- Create a mirror set (RAID1) with the name “VLBISysMon” on the existing disks

- 

- Push “Create Volume” and accept the the overwrite warning
- 

- Exit the configuration utility

- 

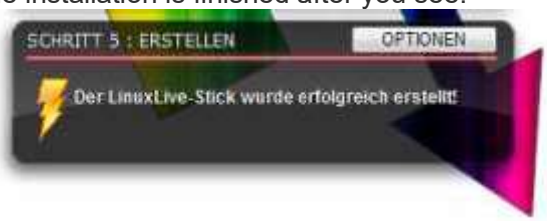
- After rebooting you should see something like this
- 

4.1.4 Download Ubuntu and install the ISO on a datastick

4.1.4.1 Methode Windows PC and LinuxLive USB Creator

- Download Ubuntu from https://www.ubuntu.com/download/desktop on a separate machine, e.g. a Windows PC
- You will get an ISO-image of the installation
- Download "**LinuxLive USB Creator**" from https://www.heise.de/download/product/linuxlive-usb-creator-90060 (do not use UNetbootin because it has some failures with 64-bit Linux/Ubuntu systems; see http://askubuntu.com/questions/544419/cant-run-a-fresh-install-of-ubuntu-14-10-shows-kernel-panick)
- Install LinuxLive USB Creator by double click on the installer program and follow the installation instructions
- Start the program LinuxLive USB Creator and create a Linux USB-stick (a detailed instruction can be found here: https://www.lidux.de/anleitungen/37-ubuntu-1210-usb-stick-installieren-creator)
  - Select the USB-stick on which the image should be installed
  - Select the ISO image of the Linux system
  - Select no "PERSISTENZ"
  - Select the formatting of the stick with FAT32
  - Click on the flash sign to start the installation

-
- The installation is finished after you see:



-
- Close the program and dismount the USB-stick

4.1.5 Methode copying the Ubuntu image on a stick

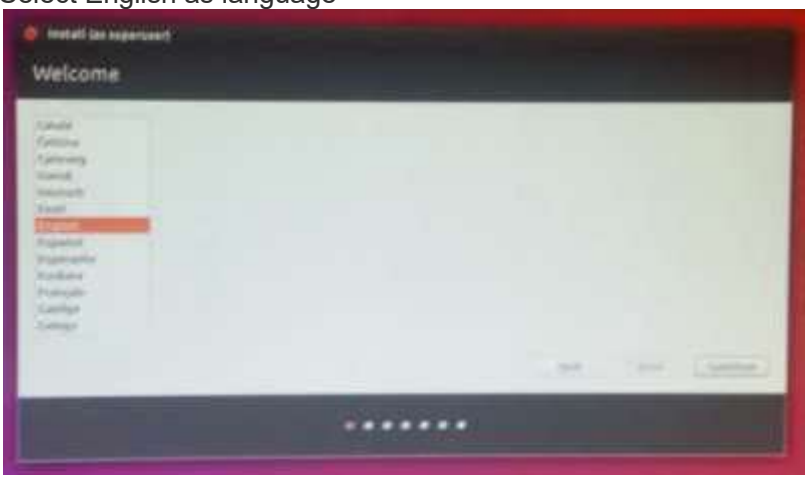
Ubuntu CD and DVD images can now be written directly to a USB stick, which is a very easy way to make a bootable USB stick.

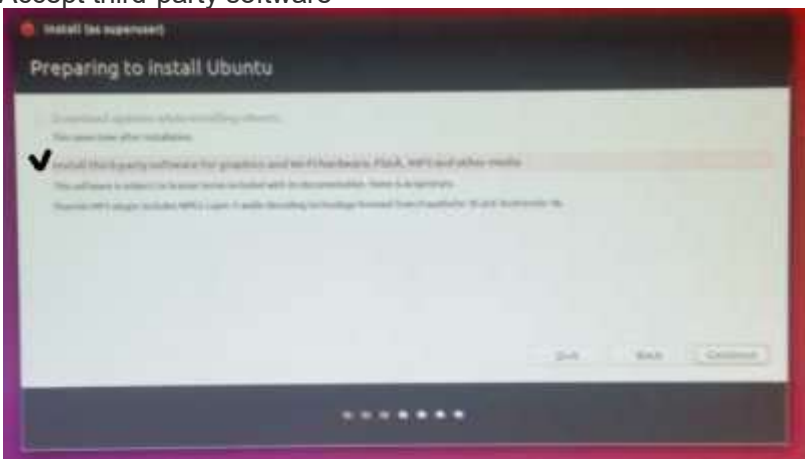
- Simply choose a CD or DVD image that will fit on your USB stick and copy it on a stick with no partitions.

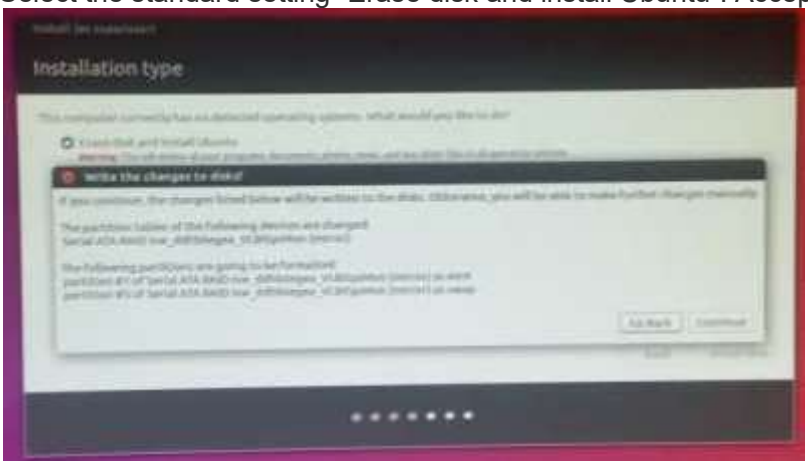
```
#sudo cp ubuntu-17.04-desktop-i386.iso /dev/sdc
```

- Further informations: https://help.ubuntu.com/16.04/installation-guide/amd64/ch04s03.html#usb-copy-isohybrid

4.1.6 Install Ubuntu on the SysMon machine

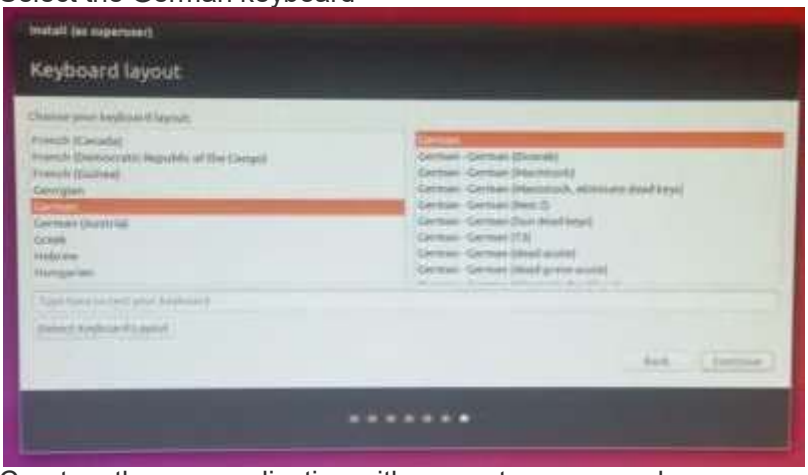
- Insert the stick into a free USB slot
- To connect keyboard and mouse, you need an USB hub, because Supermicro X7DWT just has two USB ports
- 
- Start the PC and select “Install Linux” when prompted
- Select English as language
- 

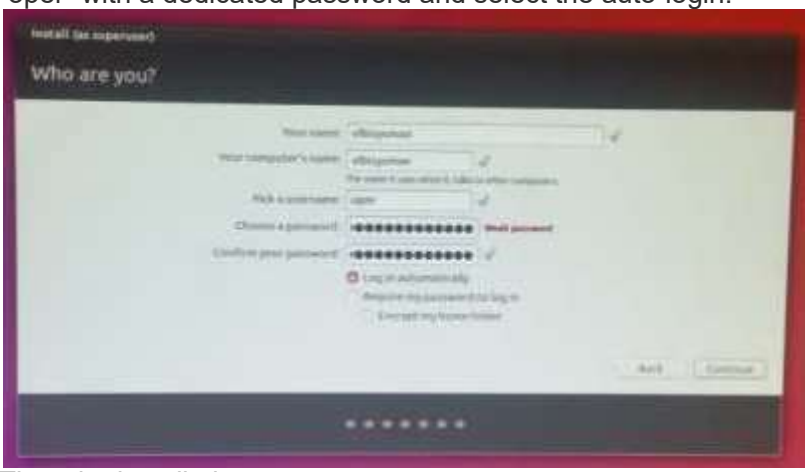
- Accept third-party software
- 

- Select the standard setting “Erase disk and install Ubuntu”. Accept the partitioning request.
- 

- Select “Berlin” as timezone by clicking onto the position of Berlin on the map

- 

- Select the German keyboard
- 

- Create a the personalization with computer name and your name as “vlbisysmon” and a user “oper” with a dedicated password and select the auto-login.
- 

- Then the installation starts

- 

- Reboot the system and keep the USB-stick in the USB-slot
- Reboot with the live system on the stick (Start Linux from the stick)
- Follow the installation of the GRUB bootloader on the RAID (see https://wiki.ubuntuusers.de/GRUB_2/Reparatur/ (German)) for a standard desktop system
  - Open a terminal (search “term” in the programs)
  - Become root

```
    sudo su
```

- Mount the RAID to /mnt

```
    mount /dev/mapper/isw_ciiaeibbja_VLBISysMon1 /mnt
    (isw stands for Intel Raid Controller; "ciiaeibbja" can be a different string)
```

- You can check the RAID (other checks can be found here: https://www.pilgermaske.org/2013/05/dmraid-mainboard-raid-unter-linux-einrichten/ (German)

```
    lsblk
```

- Mount the required directories for the GRUB installation

```
    sudo mount -o bind /dev /mnt/dev
    sudo mount -o bind /sys /mnt/sys
    sudo mount -t proc /proc /mnt/proc
    cp /proc/mounts /mnt/etc/mtab
```

- Change into root environment of the installed system on the RAID

```
    chroot /mnt /bin/bash
```

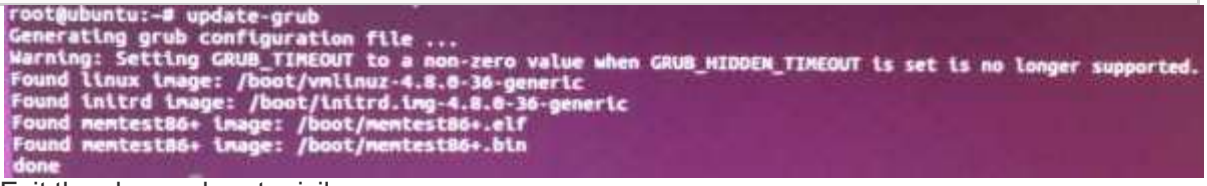
- Install GRUB (**Attention: the RAID must be used and not a partition on the RAID; this is defined by the device path without an ending number ⇒ not …_VLBISysMon1, but …_VLBISysMon**)

```
    grub-install /dev/mapper/isw_ciiaeibbja_VLBISysMon
```

- 

- Update GRUB

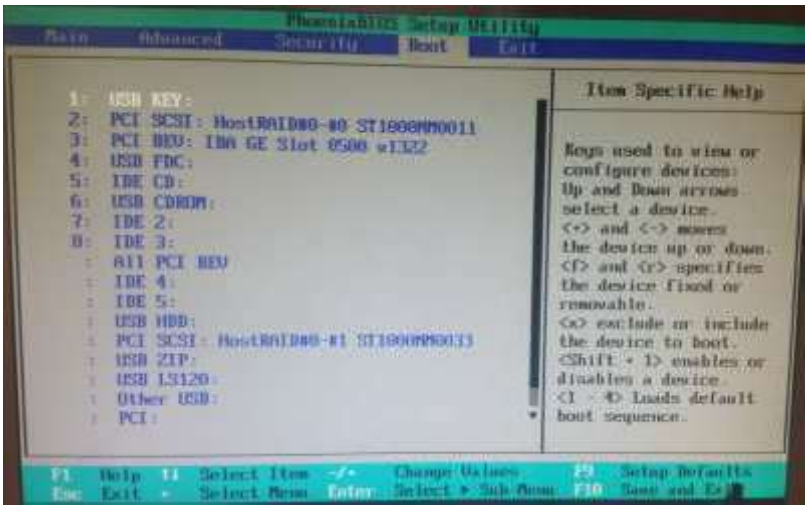
```
update-grub
```



- Exit the changed root privileges

```
exit
```

- Reboot the system and extract the USB-stick, so that the system boots from the harddrives. Check bootz order before.



- Open a terminal and become root again

```
sudo su
```

- If necessary, set a APT-proxy with "*cat > /etc/apt/apt.conf*", where the following must be entered (finish with "Ctrl+C")

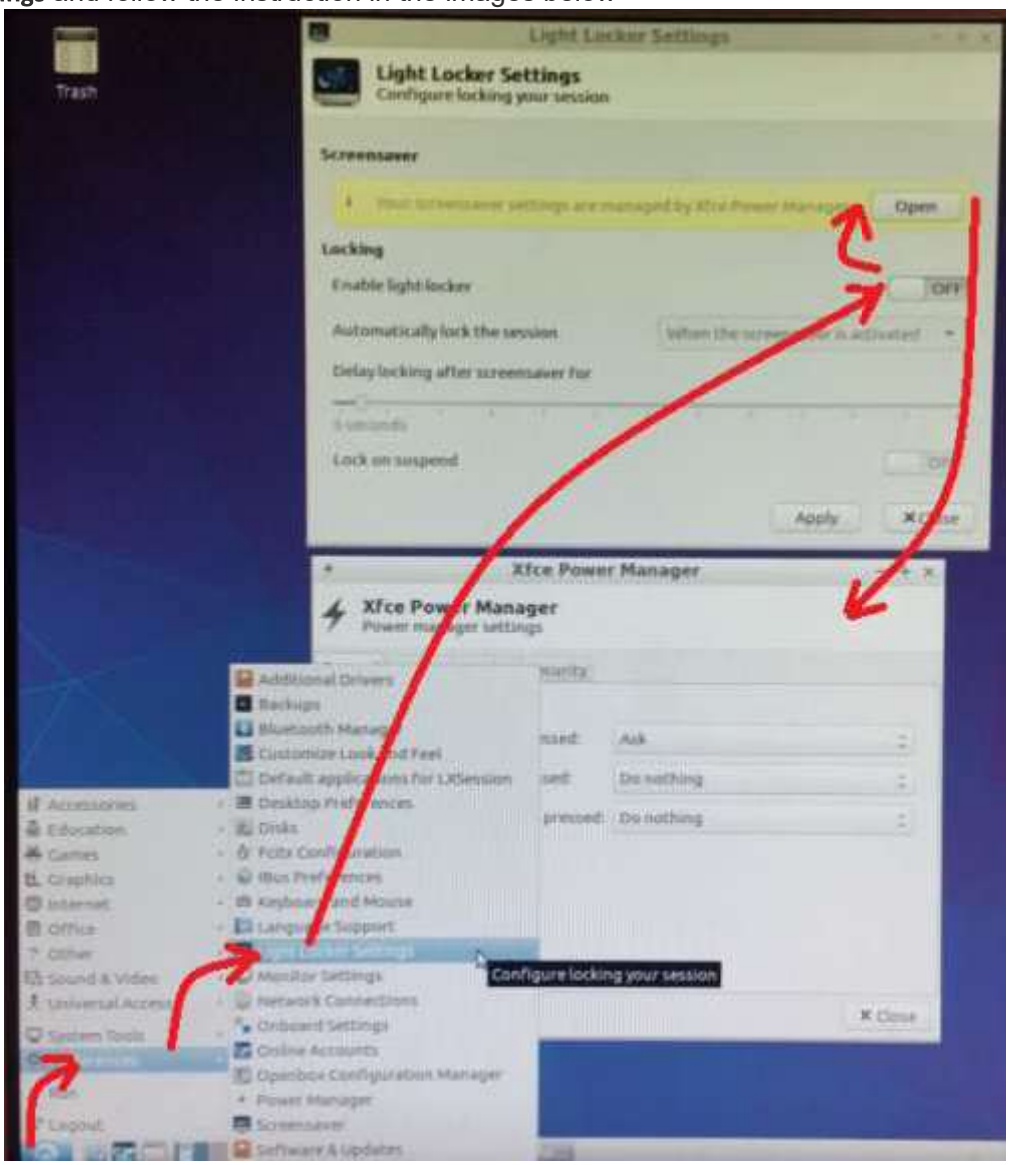
```
Acquire::http::Proxy "http://gate-w.wettzell.ifag.de:8000";
```

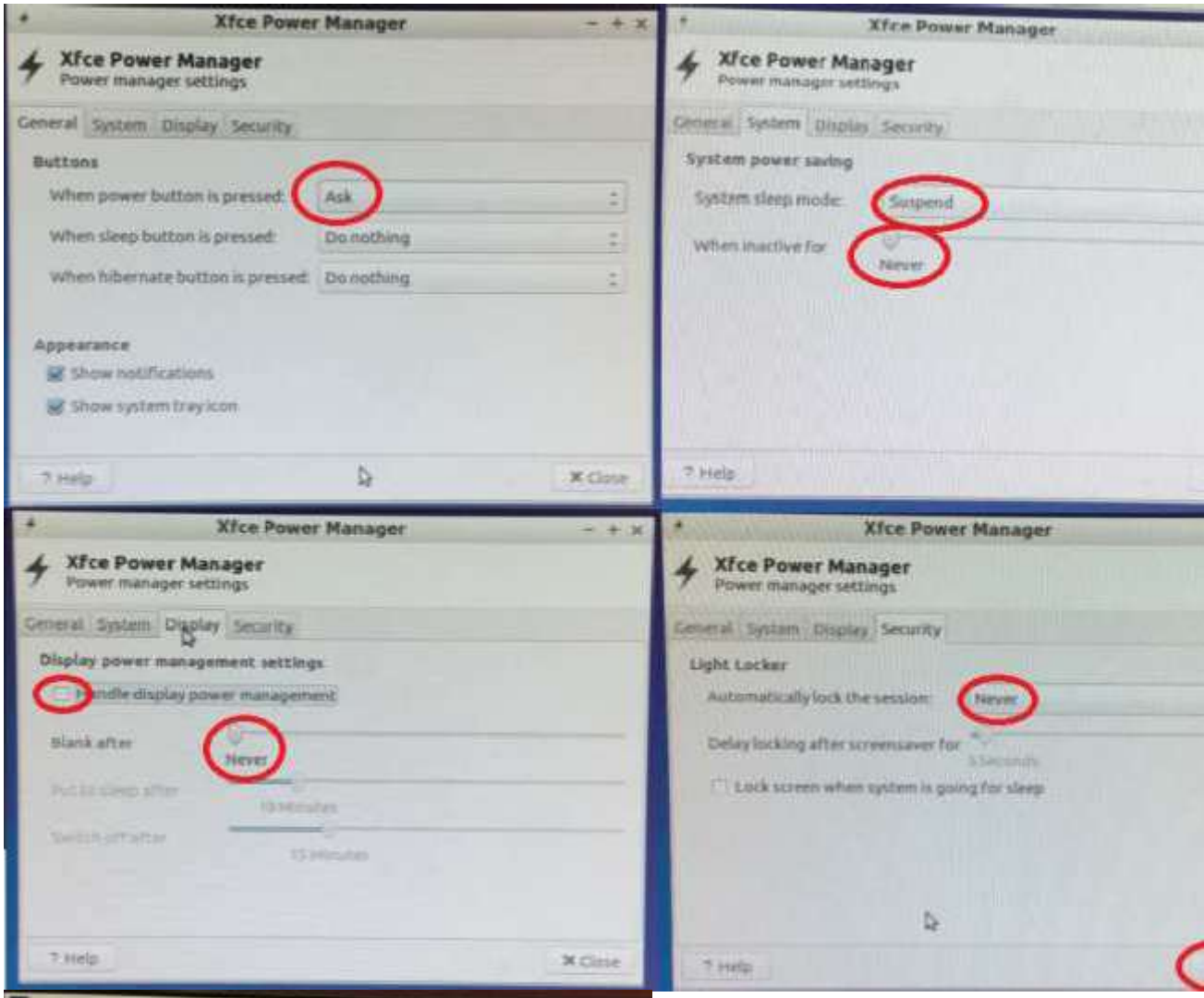
- Update package information

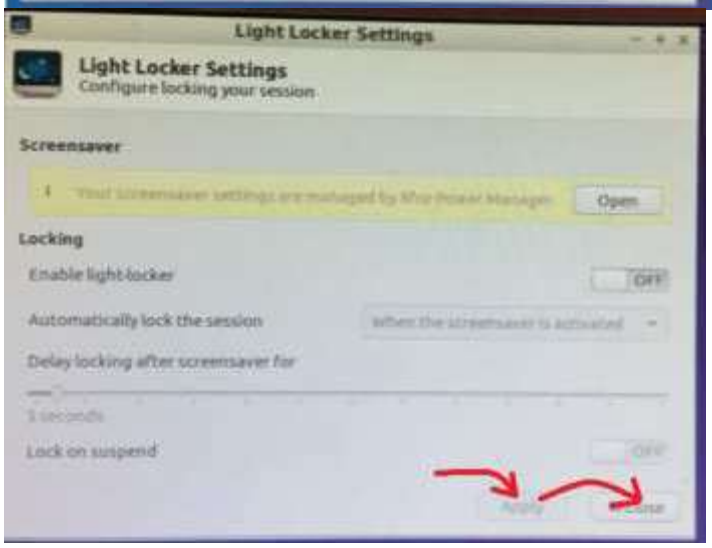
```
apt-get update
```

- **Downgrade the desktop environment from "Unity" to a lightweight one, e.g. "LXDE (Lightweight X11 Desktop Environment)"**, (it is still possible to change the environment after log-out and clicking onto the Ubuntu logo over the user login)
  - with " *sudo apt-get install lubuntu-desktop* "
  - and set it as default environment:
    - check which environments are available with " *ls /usr/share/xsessions/* " and if Lubuntu.desktop exists and
    - edit the default settings file with " *vi /usr/share/lightdm/lightdm.conf.d/50-ubuntu.conf* " as root and change it to

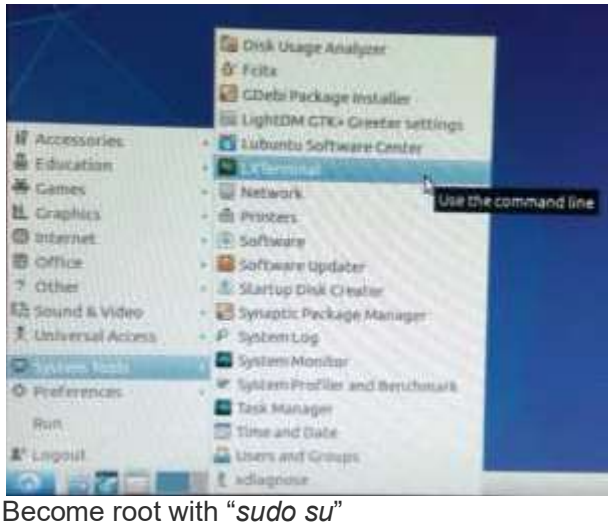
```
[SeatDefaults]
user-session=Lubuntu
```

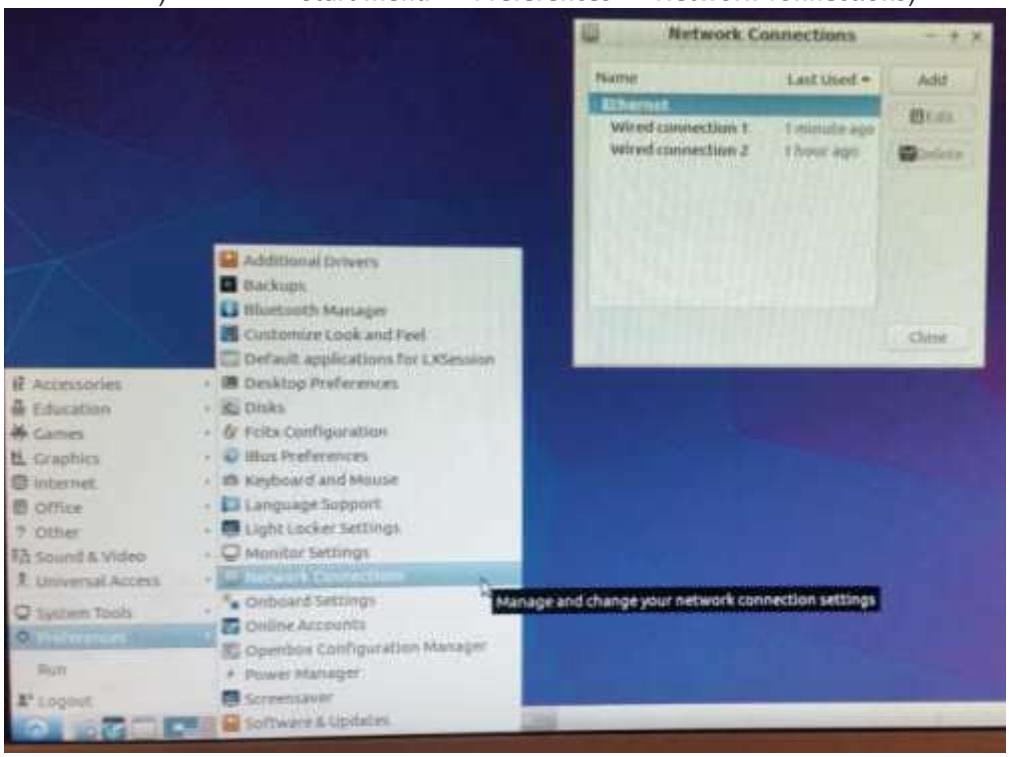
-
- Reboot
- (Maybe it is necessary to change “update-apt-xapi”-settings, which updates the software database regularly and takes a lot of CPU time)
- **Set all parameters in the screensaver** in the menu **Start menu** → **Preferences** → **Light Locker Settings** and follow the instruction in the images below
- 

- 

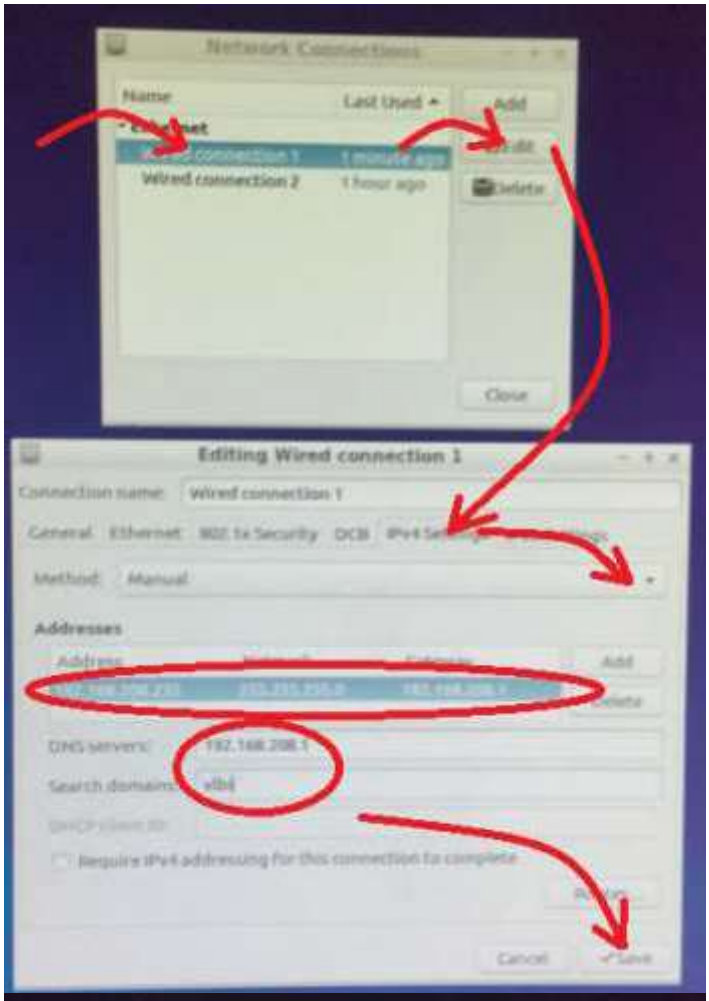
- 

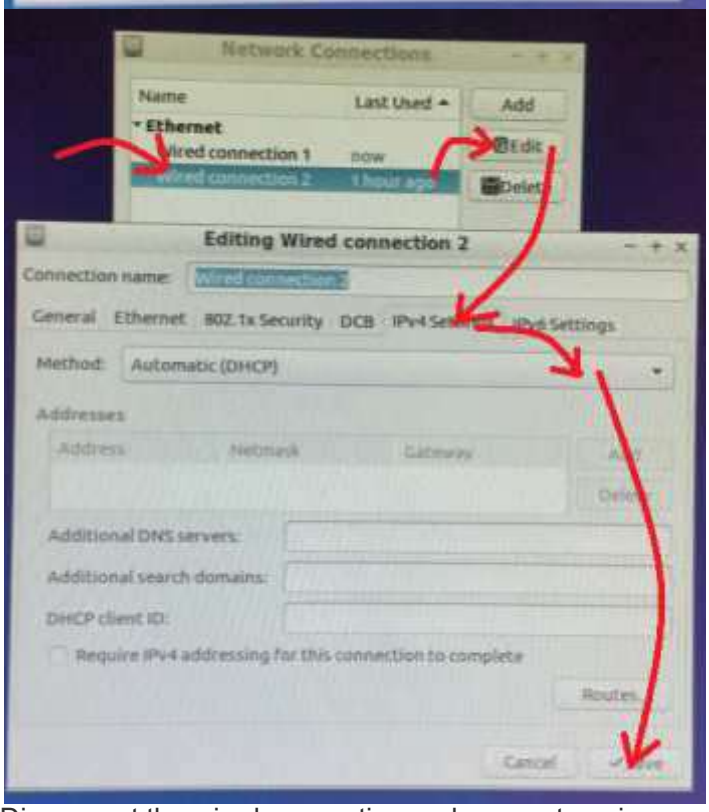
- **Set the network**
  - Open a terminal (“LXTerminal”) using the start menu

- 

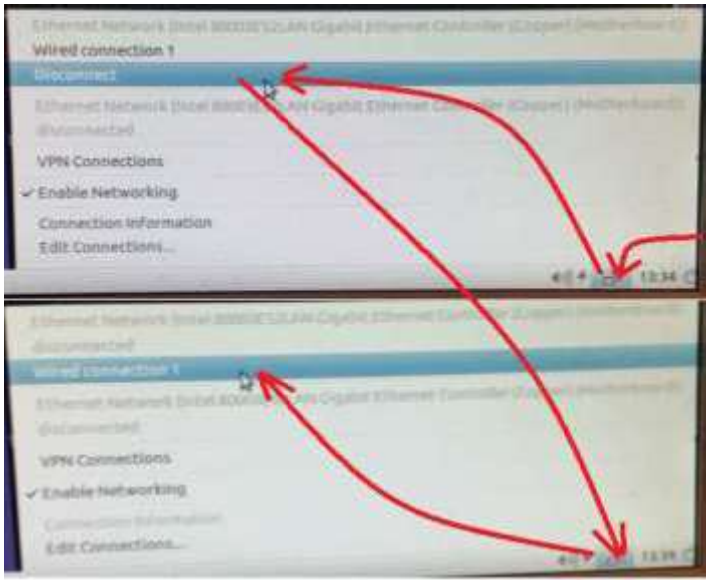
- Become root with "*sudo su*"
- Set hostname if not already correct: "*vi /etc/hostname*" and set it to "vlbisysmon"
- Open the "Network Connections" dialog (under the LXDE (Lightweight X11 Desktop Environment) it is in the **Start menu** → **Preferences** → **Network Connections**)
- 

- Each server has two network interfaces. The first one gets a static IP setting with an fixed IP from the IP-table (see IP-addresses of the "vlbi" network) and the second gets a DHCP setting (this can be let as it is in the standard installation)

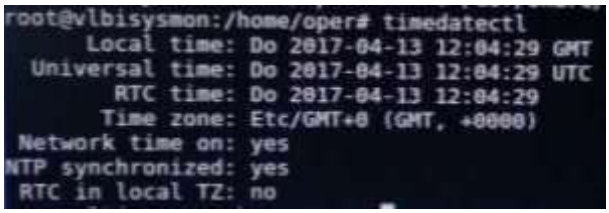
- 

- 

- Disconnect the wired connection and connect again

- 

- Check the correct settings using "*ifconfig*" in a terminal
- 

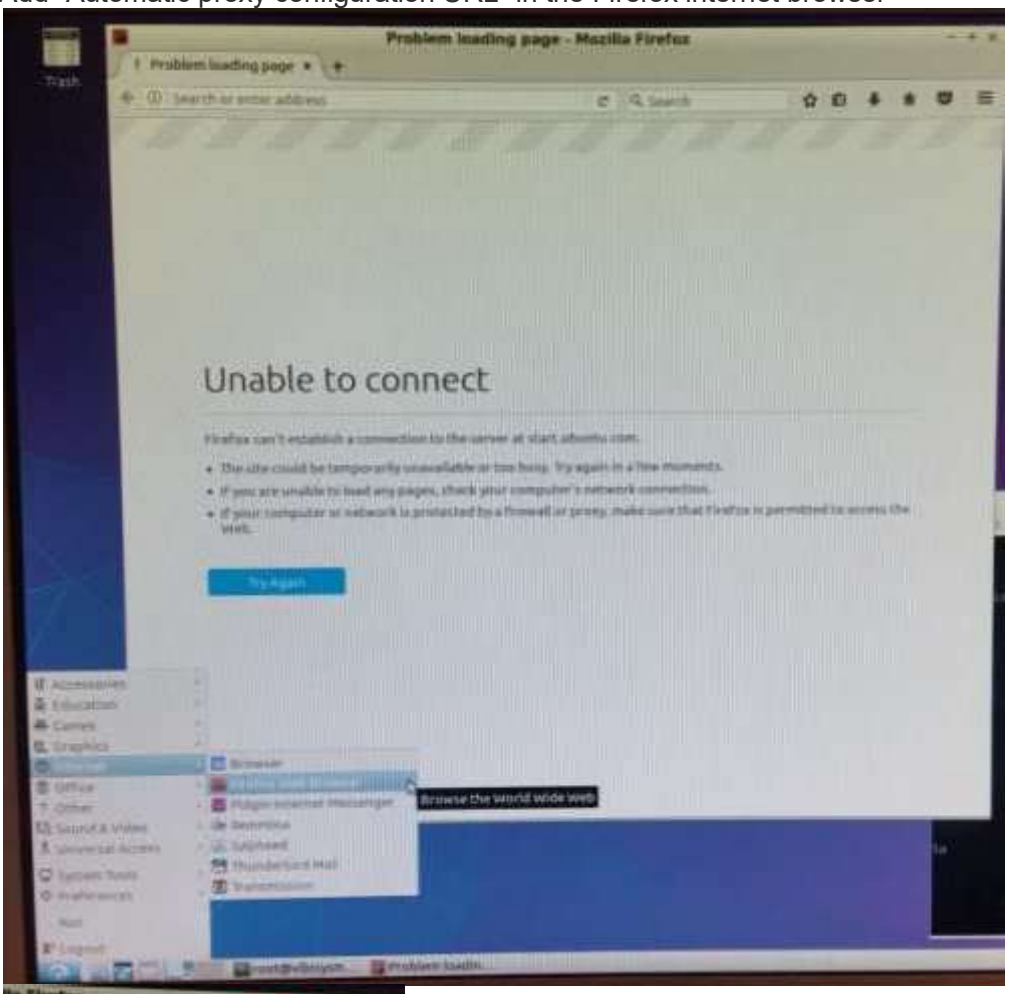
- **Set clock to UTC (GMT+0)**
  - First set the time and timezone in the desktop with **Start menu → System tools → Time and Date**
  - Set hardware clock to UTC (as user "root"): "*vi /etc/default/rcS*" and set line "*UTC=yes*"
  - Run "*timedatectl set-local-rtc 0* "
  - Set localtime to GMT+0: "*rm /etc/localtime*" and "ln -s /usr/share/zoneinfo/Etc/GMT+0 /etc/localtime"
  - Check it with "*timedatectl*". You should see something like this:
  - 

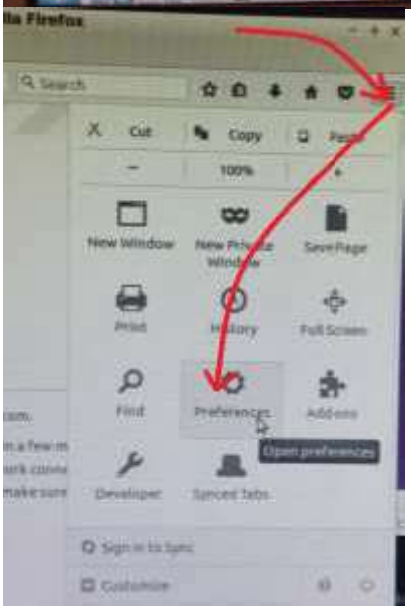
- **Activate NTP**
  - "*apt-get install ntp*"
  - "*apt-get install ntpdate*"
  - Set local NTP servers for "ntpdate": "*vi /etc/default/ntpdate*"
    - Set line "*NTPSERVERS= "192.168.208.4 192.168.208.5"*" (delete the existing NTPSERVERS line)
  - Set local NTP servers for "ntpd": "*vi /etc/ntp.conf*"
    - Set line "*server 192.168.208.4*"
    - Set line "*server 192.168.208.5*"
    - Set all existing server lines as comments (starting '#')
    - Set all existing pool lines as comments (starting '#')
  - Set current time once
    - " */etc/init.d/ntp stop*"
    - "*ntpdate -s 192.168.208.4*"

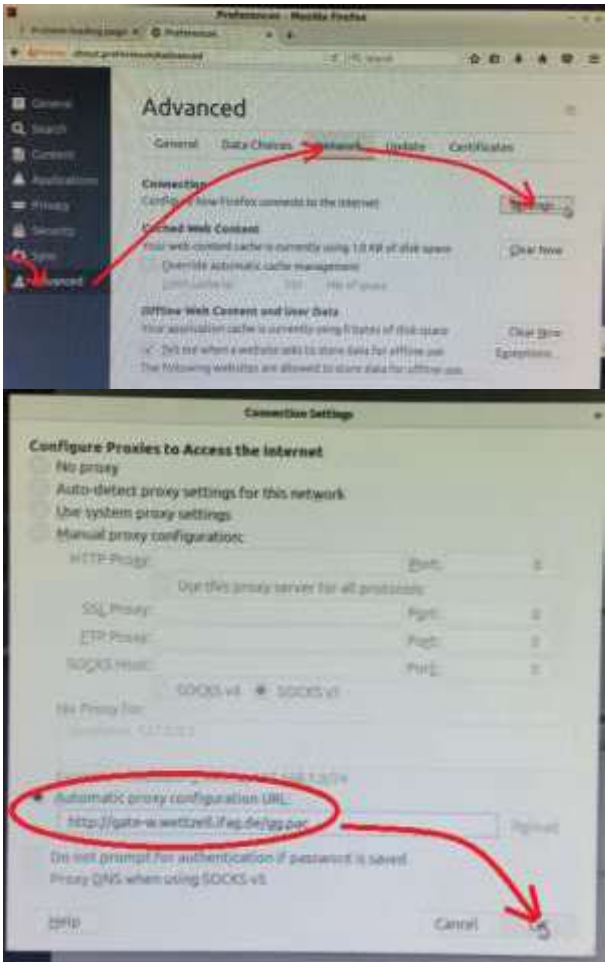
- " */etc/init.d/ntp start*"
- Check NTP status
  - " *ntpq -p*"

4.1.7 Costomize Linux software for system monitoring

4.1.7.1 Firefox browser

- Add "Automatic proxy configuration URL" in the Firefox internet browser
- 

- 

- 

-
- If another browser should also be used, do the same setting there.

4.1.8 SSH server

- Install a SSH server with "*apt-get install ssh*" (or as minimum "*apt-get install openssh-server*")
- Install "autossh" to automatically restart SSH sessions and tunnels with "*apt-get install autossh*"
- **Hint: Getting X11 forwarding through ssh working after running su**
  - Run "*xauth list $DISPLAY* " to get the cookie of the SSH connection, e.g. "*somehost.somedomain:10 mit-magic-cookie-1 4d22408a71a55b41ccd1657d377923ae* "
  - Change user with "*sudo su*"
  - Run "*xauth add «<cookie»* ", e.g. "*xauth add somehost.somedomain:10 mit-magic-cookie-1 4d22408a71a55b41ccd1657d377923ae* " to add the forwarding cookie to the new user
  - Create an SSH key for user "oper", using "*ssh-keygen -b 4096* " and save it to file "*vlbisysmonoper*" (vlbisysmonoper.zip)

```
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): vlbisysmonroot
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in vlbisysmonroot.
Your public key has been saved in vlbisysmonroot.pub.
The key fingerprint is:
SHA256:q99jHH06xLDpH7PMDZPCdvjr8zCqQV10048ZW/+esiI root@vlbisysmon
The key's randomart image is:
+---[RSA 4096]----+
|            . o. |
|           . ...o|
|            .  *o|
|         ...  + o|
|        S .*    .|
|       . o+.+.. .|
|        oo=oXo ..|
|       . E=O+@ ..|
|      ..oo++X*=  |
+----[SHA256]-----+
```

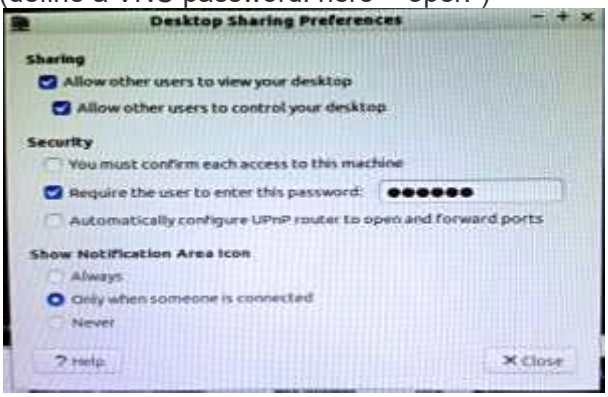
- Install the new key file "*ssh-copy-id -i vlbisysmonoper.pub oper@192.168.208.236* "
- The new key is installed at " */home/oper/.ssh/authorized_keys* "
- Permit password authenication by editing " */etc/ssh/sshd_config* " and activate the following line with "no"

```
51    # Change to no to disable tunnelled clear text passwords
52    PasswordAuthentication no
```

- Restart ssh daemon with " */etc/init.d/ssh restart*"
- From now on login is only possible using "*ssh -X -i vlbisysmonoper oper@192.168.208.236* "
- Note: If you want to use the key with Putty on Windows, you have to use the program "puttygen" to convert the key to a *.ppk file in the Putty format. Open "puttygen" and follow the menu "File" ⇒ "Load private key" and open the private key generated before. It converts the key. Save the new key by pushing on the button "Save private key" and store it as "vlbisysmonoper.ppk". Then open Putty and create a new connection. Open the menu "SSH" ⇒ "Auth" and add the new private key in *.ppk format.

4.1.8.1 Vino VNC server

- Configure the "Desktop Sharing Preferences" by calling "*vino-preferences*" as user "oper" (define a VNC password: here "+oper!")



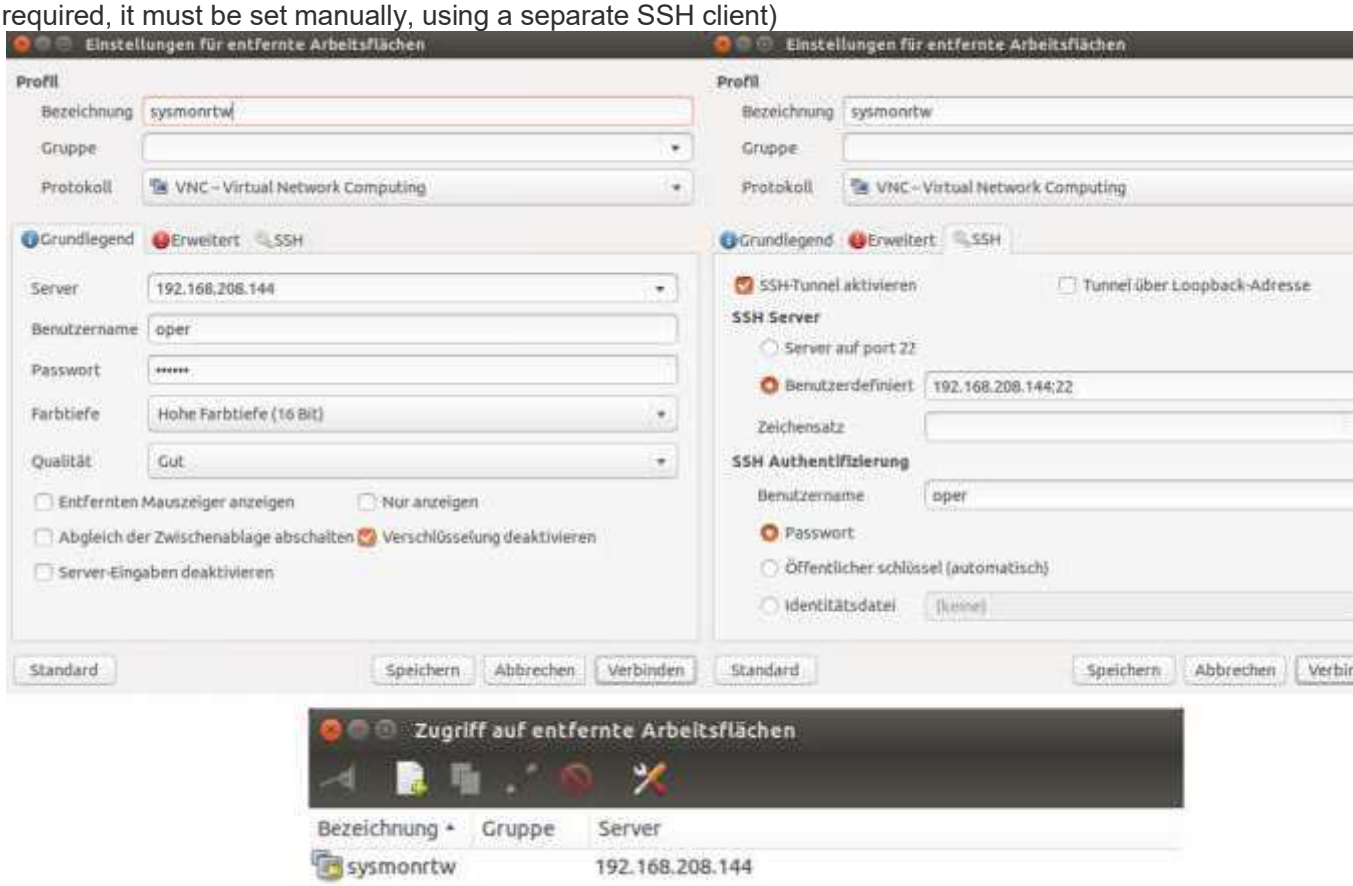
- Disable encryption, to easily allow the access with all VNC clients, with "*gsettings set org.gnome.Vino require-encryption false*"
- Create a new directory (if not yet available) as user "oper": "*mkdir /home/oper/Software* " and "*mkdir /home/oper/Software/vino_vnc* "
- Change into the new directory with "*cd /home/oper/Software/vino_vnc* "
- Create a start script "vinovnc.sh" with an editor in the new folder and add the following content:

```
#!/bin/bash
/usr/lib/vino/vino-server > /dev/null 2> /dev/null &
```

- Change the access rights of the new script with “*chmod 744 ./vinovnc.sh* ”
- Create a desktop starter file with “*vi /home/oper/.config/autostart/vinovnc.desktop* ” and add the following context (it can also be created with the program “lxshortcut”):

```
[Desktop Entry]
Type=Application
Name=Vino VNC server
Comment=Automatic start of the VINO VNC server
Exec=/home/oper/Software/vino_vnc/vinovnc.sh
Terminal=false
```

- Test the automatic start: log-out and -on again, which should start the application (test it with “*ps ax | grep vino*”)
- The VNC Ports are: **5800** and **5900**
- An example configuration of a remote VNC client can look like the following setup for the Ubuntu “Remmina Remote Desktop Client” (similar settings can also be used for other VNC clients, like “Real VNC” under windows or xvnc4viewer under Linux; just if a tunneling is required, it must be set manually, using a separate SSH client)



-

4.1.8.2 Editor geany

- Install geany using the command “*apt-get install geany*”

4.1.8.3 GNU g++ compiler

- Install g++ using the command “*apt-get install g++*”

4.1.8.4 Subversion

- Install Subvserion as root with “*apt-get install subversion*”

4.1.8.5 PostgreSQL 9.5

- “*apt-get install postgresql-9.5*”

- The PostgreSQl database is then at " */var/lib/postgresql/9.5/main*"
- The PostgreSQL configuration is then at " */etc/postgresql/9.5/main/postgresql.conf*" (to find the current location of the configuration file use "*ps ax | grep postgres*", which prints the complete calling arguments of the server including the "config_file" parameter, e.g. "*/usr/lib/postgresql/9.5/bin/postgres -D /var/lib/postgresql/9.5/main -c config_file=/etc/postgresql/9.5/main/postgresql.conf* ")
- Enable remote access
  - "*vi /etc/postgresql/9.5/main/postgresql.conf*" and enable "*listen_addresses = 'localhost'*" and "*port = 5432*"
  - "*vi /etc/postgresql/9.5/main/pg_hba.conf*" and enable "host all all 127.0.0.1/32 trust"

```
# Database administrative login by UNIX sockets
local   all         postgres                          trust
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD
# "local" is for Unix domain socket connections only
local   all         all                               trust
# IPv4 local connections:
host    all         all         127.0.0.1/32          trust
# IPv6 local connections:
host    all         all         ::1/128               trust
# Zabbix database access
local   zabbix      zabbix                            md5
```

- Restart PostgreSQL with " */etc/init.d/postgresql stop*" and " */etc/init.d/postgresql start*" ("*/etc/init.d/postgresql-8.4 restart*" my not work correctly
- Test the connectivity with " *psql -h 127.0.0.1 -p 5432 postgres postgres*" (quit with Ctrl-D)
- For the programming simple_psqlquery can be used
- Further documentation can be found on http://www.postgresql.org/docs/9.5/static/index.html
- Install the PostgreSQL library for the compiler using "*apt-get install libpq-dev*"

4.1.9 Wettzell System Monitoring Software (SysMon)

- The software can be found on the Wettzell Subversion repository http://xsamba.wtz/svn/vlbi/trunk/code/vlbisysmon/
- Create a directory "Software" in the home directory of the user oper with "*mkdir /home/oper/Software* "
- Change into the new directory and fetch the SysMon source with the Subversion command "*svn co http://xsamba.wtz/svn/vlbi/trunk/code/vlbisysmon/* "
- Connect to PostgreSQL using " *psql -h 127.0.0.1 -p 5432 postgres postgres*" (quit with Ctrl-D)
- Create role and database:
  - "*CREATE ROLE sysmon ENCRYPTED PASSWORD '+sysmon!' SUPERUSER NOCREATEDB NOCREATEROLE NOINHERIT LOGIN CONNECTION LIMIT 100;*"
  - "*CREATE DATABASE sysmon WITH OWNER=sysmon;*"
- Test the connectivity to the new database with " *psql -h 127.0.0.1 -p 5432 sysmon sysmon*" (quit with Ctrl-D)
- Change into directory of Wettzell SysMon software and build the individual components which you want to use

```
cd /home/oper/Software/vlbisysmon/main/sysmon_sender/make
make build
cd /home/oper/Software/vlbisysmon/main/sysmon_backup/make
make build
```

4.1.10 Apache web server

- Install Apache2 as root with "*apt-get install apache2*"

4.1.11 PHP

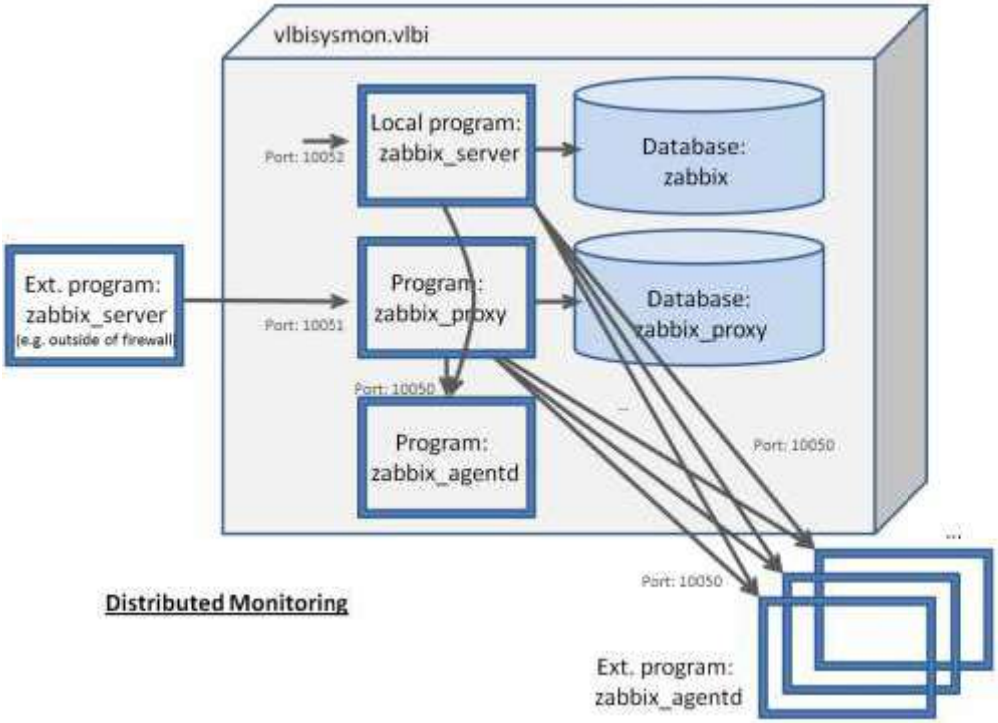
- "*apt-get install php libapache2-mod-php php-mcrypt*"

4.1.12 automake

- "*apt-get install automake*"

4.1.12.1 Zabbix

- Idea of a distributed monitoring concept (the map as Powerpoint file)



-
- A basic manual can be found here: https://www.zabbix.com/documentation/2.2/manual
- Install the Zabbix software using
  - *"apt-get install zabbix-server-pgsql"*
  - *"apt-get install zabbix-agent"*
  - *"apt-get install zabbix-frontend-php"*
- Create log file folders
  - *"mkdir /var/log/zabbix-server"*
  - *"mkdir /var/log/zabbix-agent"*
  - *"mkdir /var/log/zabbix-proxy"*
  - *"chown zabbix:zabbix /var/log/zabbix-server"*
  - *"chown zabbix:zabbix /var/log/zabbix-agent"*
  - *"chown zabbix:zabbix /var/log/zabbix-proxy"*
- Configure the server with *"geany /etc/zabbix/zabbix_server.conf"*

```
ListenPort=10052
DBHost=localhost
DBName=zabbix
DBUser=zabbix
DBPassword=zabbix
LogFile=/var/log/zabbix-server/zabbix_server.log
```

- Create the zabbix database after connecting with " psql -h 127.0.0.1 -p 5432 postgres postgres" (quit with Ctrl-D)

```
CREATE USER zabbix WITH PASSWORD 'zabbix';
CREATE DATABASE zabbix OWNER zabbix;
```

- Create the zabbix proxy database after connecting with " psql -h 127.0.0.1 -p 5432 postgres postgres" (quit with Ctrl-D)

```
CREATE USER zabbix_proxy WITH PASSWORD 'zabbix_proxy';
CREATE DATABASE zabbix_proxy OWNER zabbix_proxy;
```

- Download the Zabbix sources which fit to the Zabbix installation of the operating system: e.g. for Ubuntu 16.04. LTS it is Zabbix 2.4.7 (to check, start "*zabbix_server*" with "*DebugLevel=3*" in the configuration file " */etc/zabbix/zabbix_server.conf* " and read the log file at " */var/log/zabbix-server/zabbix_server.log* ", which is also defined in the configuration file of the server):
  - zabbix_3.2.4.orig.tar.gz
  - or download from http://www.zabbix.com/download.php to the directory /home/oper/Software/ and extract the package with "*tar -zxvf zabbix_3.2.4.orig.tar.gz*"

- Change into folder /home/oper/Software/zabbix-3.2.4/database/postgresql and run (see https://www.zabbix.com/documentation/3.2/manual/appendix/install/db_scripts) for the server

```
psql -h 127.0.0.1 -p 5432 -U zabbix zabbix < schema.sql
# stop here if you are creating database for Zabbix proxy
psql -h 127.0.0.1 -p 5432 -U zabbix zabbix < images.sql
psql -h 127.0.0.1 -p 5432 -U zabbix zabbix < data.sql
```

- and for the proxy

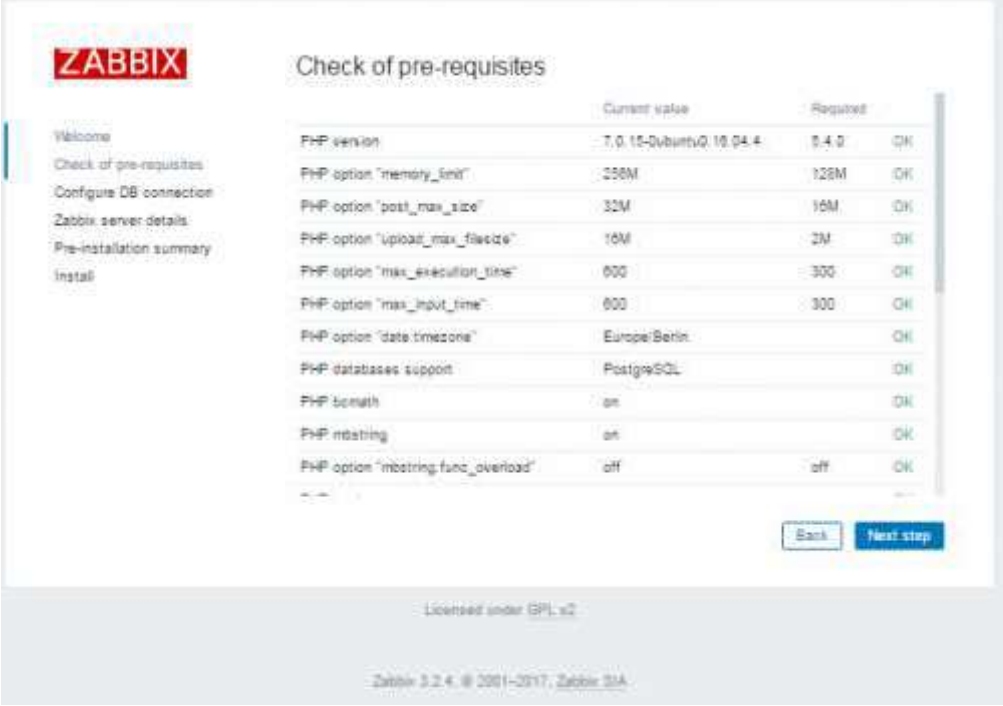
```
psql -h 127.0.0.1 -p 5432 -U zabbix_proxy zabbix_proxy < schema.sql
```

- Restart Zabbix server process
  - " */etc/init.d/zabbix-server stop* "
  - " */etc/init.d/zabbix-server start* "
- Configure PHP with "*geany /etc/php/7.0/apache2/php.ini* " and restart the Apache2 server with "*/etc/init.d/apache2 stop* " and " */etc/init.d/apache2 start*"

```
[Date]
; Defines the default timezone used by the date functions
date.timezone = Europe/Berlin
max_execution_time = 600
post_max_size = 32M
memory_limit = 256M
mbstring.func_overload = 0
upload_max_filesize = 16M
max_input_time = 600
```

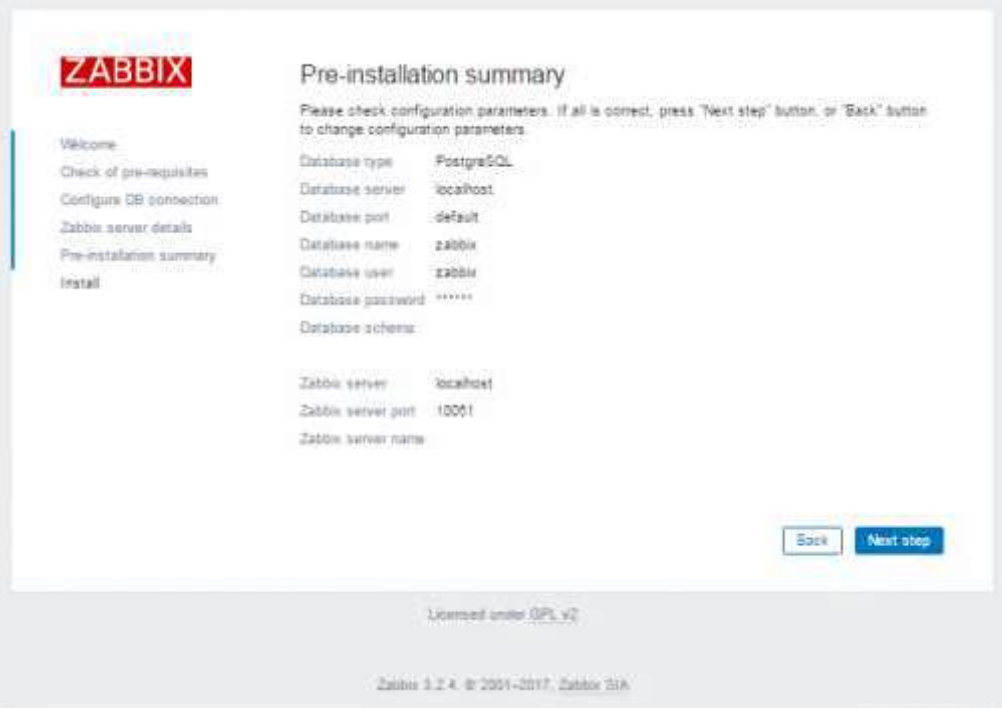
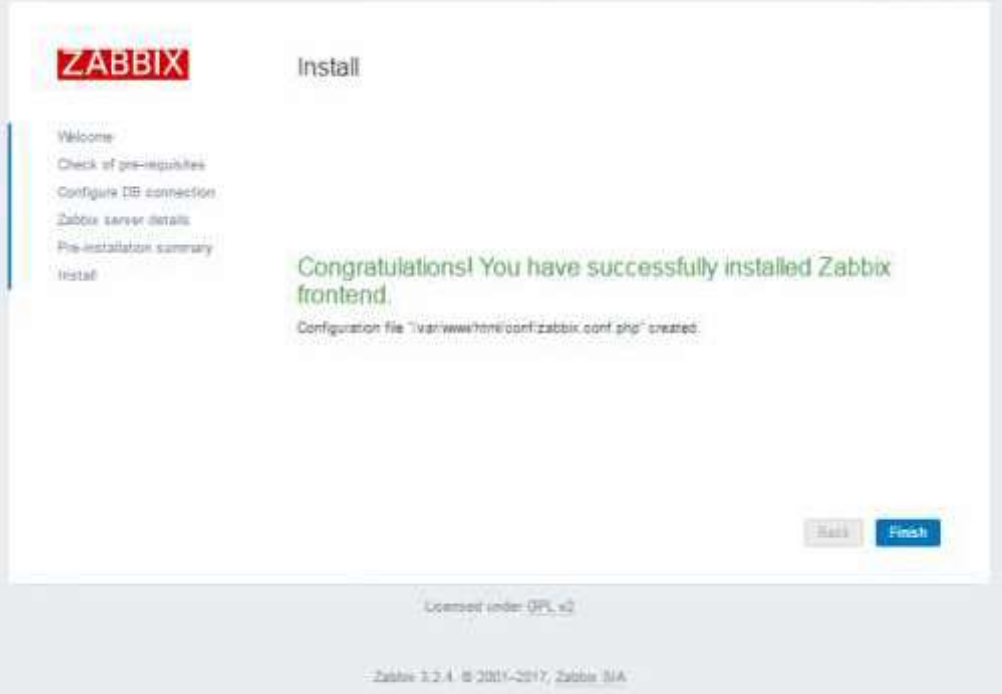
- Create Web front-end as root
  - " *cd /var/www* "
  - " *mv /var/www/html/ /var/www/html_original* "

- " *chown -R www-data:www-data /var/www/html_original* "
- " *mkdir html* "
- " *cp -R /home/oper/Software/zabbix-3.2.4/frontends/php/* ./html/.* "
- " *chown -R www-data:www-data /var/www/html* "
- Restart the Apache2 server with " */etc/init.d/apache2 stop* " and " */etc/init.d/apache2 start*"
- Open a browser and connect to "http://127.0.0.1" and follow the instructions (if the configuration file cannot be saved automatically, then download it and save it at /var/www/html/conf/.
- 

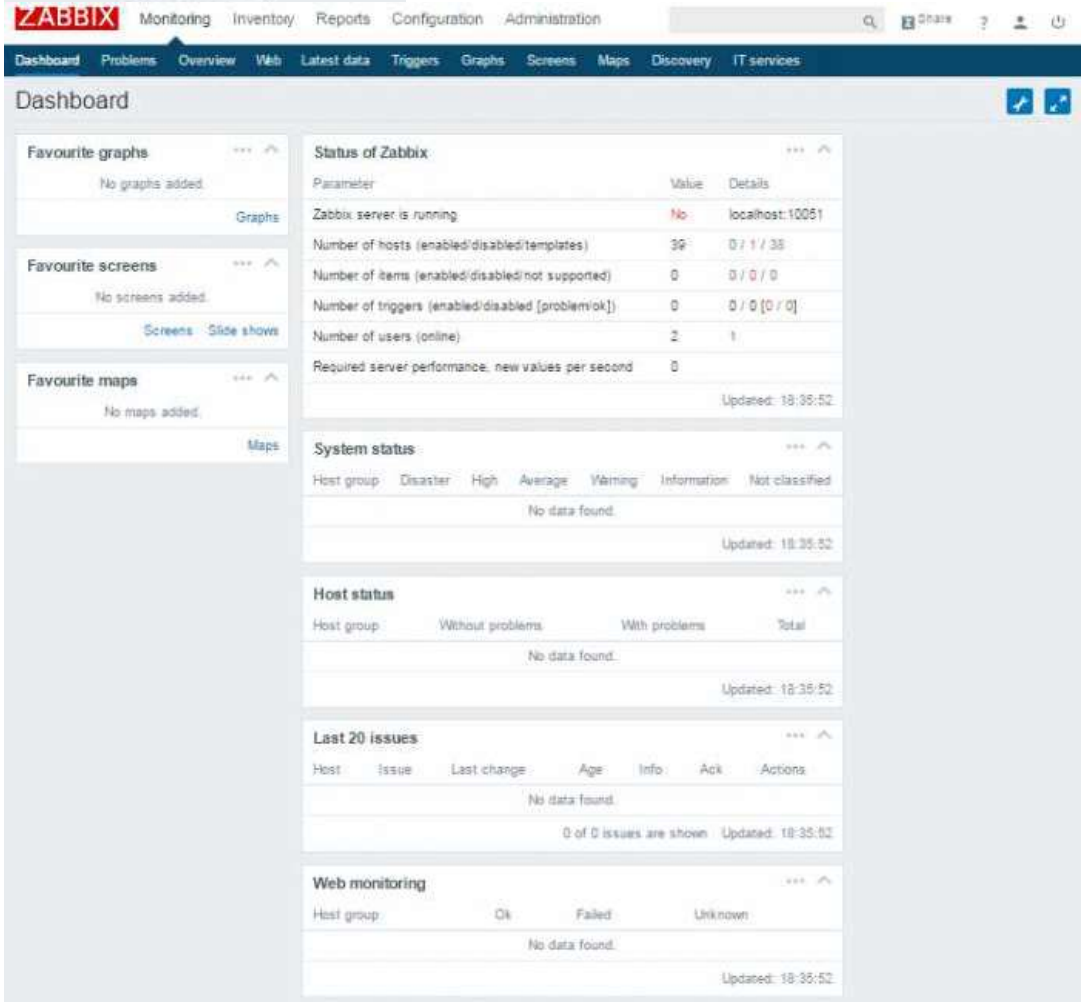
- 

- 
ZABBIX
Configure DB connection
Please create database manually, and set the configuration parameters for connection to this database. Press "Next step" button when done.
Welcome
Check of pre-requisites
Configure DB connection
Zabbix server details
Pre-installation summary
Install
Database type
PostgreSQL
Database host
localhost
Database port
0
0 - use default port
Database name
zabbix
User
zabbix
Password
zabbix
Back
Next step
Licensed under GPL v2
Zabbix 3.2.4. © 2001–2017, Zabbix SIA

- 
ZABBIX
Zabbix server details
Please enter the host name or host IP address and port number of the Zabbix server, as well as the name of the installation (optional).
Welcome
Check of pre-requisites
Configure DB connection
Zabbix server details
Pre-installation summary
Install
Host
localhost
Port
10051
Name
Back
Next step
Licensed under GPL v2
Zabbix 3.2.4. © 2001–2017, Zabbix SIA

- 
ZABBIX
Pre-installation summary
Please check configuration parameters. If all is correct, press "Next step" button, or "Back" button to change configuration parameters.
Welcome
Check of pre-requisites
Configure DB connection
Zabbix server details
Pre-installation summary
Install
Database type PostgreSQL
Database server localhost
Database port default
Database name zabbix
Database user zabbix
Database password ******
Database schema
Zabbix server localhost
Zabbix server port 10051
Zabbix server name
Back
Next step
Licensed under GPL v2
Zabbix 3.2.4. © 2001-2017, Zabbix SIA
- 
ZABBIX
Install
Welcome
Check of pre-requisites
Configure DB connection
Zabbix server details
Pre-installation summary
Install
Congratulations! You have successfully installed Zabbix frontend.
Configuration file "/var/www/html/conf/zabbix.conf.php" created.
Back
Finish
Licensed under GPL v2
Zabbix 3.2.4. © 2001-2017, Zabbix SIA

- 

- 

-
- Prepare manual Zabbix installation

```
    sudo apt-get install libsnmp-dev
```

- Update the “zabbix_server” and “zabbix_agent” to the latest version
  - Change into directory “ */home/oper/Software/zabbix-3.2.4/* ”
  - Run a configuration

```
./configure --enable-server --enable-agent --enable-proxy --with-postgresql --with-net-snmp
```

- Build the server and agent

```
make
```

- Copy server, agent and proxy to /usr/sbin

```
mv /usr/sbin/zabbix_server /usr/sbin/zabbix_server_2.4.7
cp /home/oper/Software/zabbix-3.2.4/src/zabbix_server/zabbix_server /usr/sbin/zabbix_server
mv /usr/sbin/zabbix_agentd /usr/sbin/zabbix_agentd_2.4.7
cp /home/oper/Software/zabbix-3.2.4/src/zabbix_agent/zabbix_agentd /usr/sbin/zabbix_agentd
cp /home/oper/Software/zabbix-3.2.4/src/zabbix_proxy/zabbix_proxy /usr/sbin/zabbix_proxy
```

- Create a new configuration file for the proxy

```
cp /home/oper/Software/zabbix-3.2.4/conf/zabbix_proxy.conf /etc/zabbix/.
```

- and edit it with "*geany /etc/zabbix/zabbix_proxy.conf* "

```
DBHost=localhost
DBName=zabbix_proxy
DBUser=zabbix_proxy
DBPassword=zabbix_proxy
ProxyMode=1 # Passive => Server fetches data
LogFile=/var/log/zabbix-proxy/zabbix_proxy.log
```

- Create a soft-link to the original configuration files

```
ln -s /etc/zabbix/zabbix_server.conf /usr/local/etc/zabbix_server.conf
ln -s /etc/zabbix/zabbix_agentd.conf /usr/local/etc/zabbix_agentd.conf
ln -s /etc/zabbix/zabbix_proxy.conf /usr/local/etc/zabbix_proxy.conf
```

- Create a shell script to test startup of server using "*geany /usr/sbin/zabbix_server.sh* " and change the mode to allow the execution of the script

```
#!/bin/bash

/usr/sbin/zabbix_server -c /etc/zabbix/zabbix_server.conf
```

- "*mode 755 /usr/sbin/zabbix_server.sh* "
- Create a shell script to test startup of proxy using "*geany /usr/sbin/zabbix_proxy.sh* " and change the mode to allow the execution of the script

```
#!/bin/bash

/usr/sbin/zabbix_proxy -c /etc/zabbix/zabbix_proxy.conf
```

- "*mode 755 /usr/sbin/zabbix_proxy.sh* "

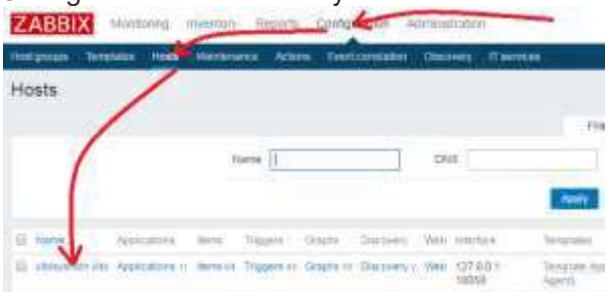
- Stop agent and server

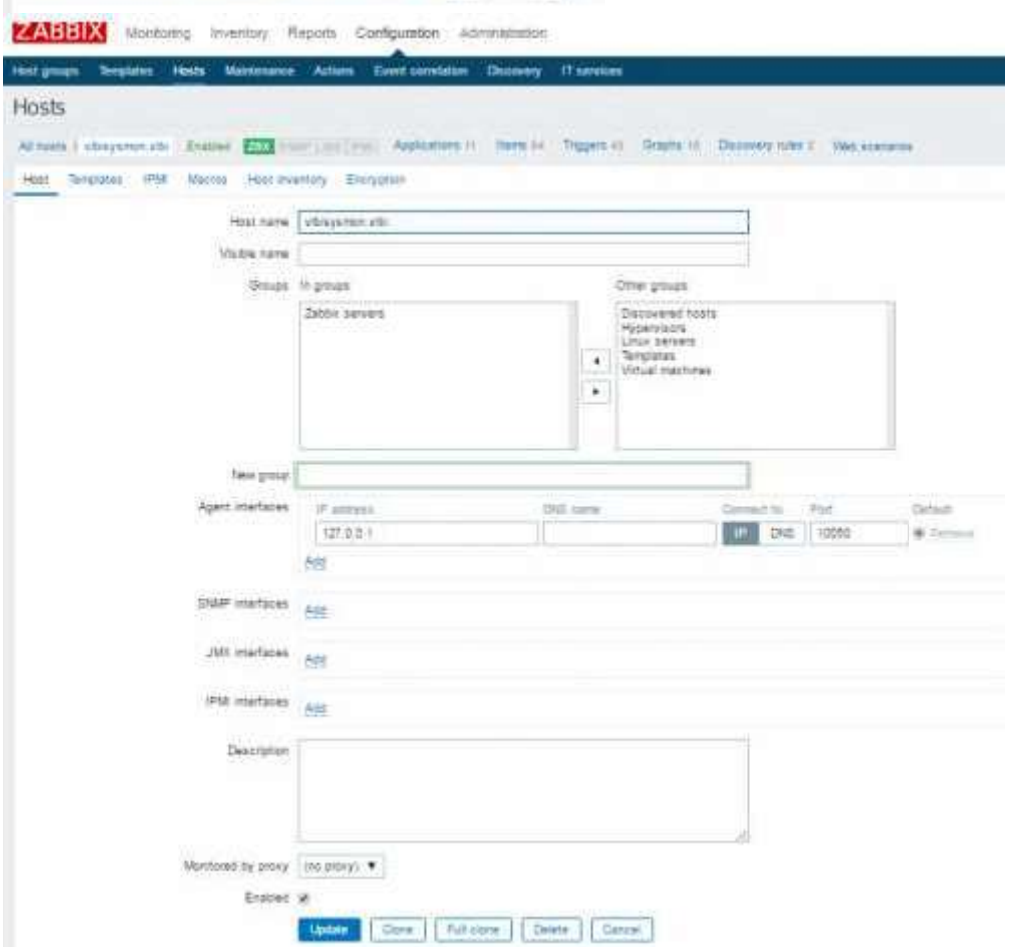
```
/etc/init.d/zabbix-server stop
/etc/init.d/zabbix-server start
/etc/init.d/zabbix-agent stop
/etc/init.d/zabbix-agent start
```

- Change server name using “*geany /etc/zabbix/zabbix_agentd.conf* ”

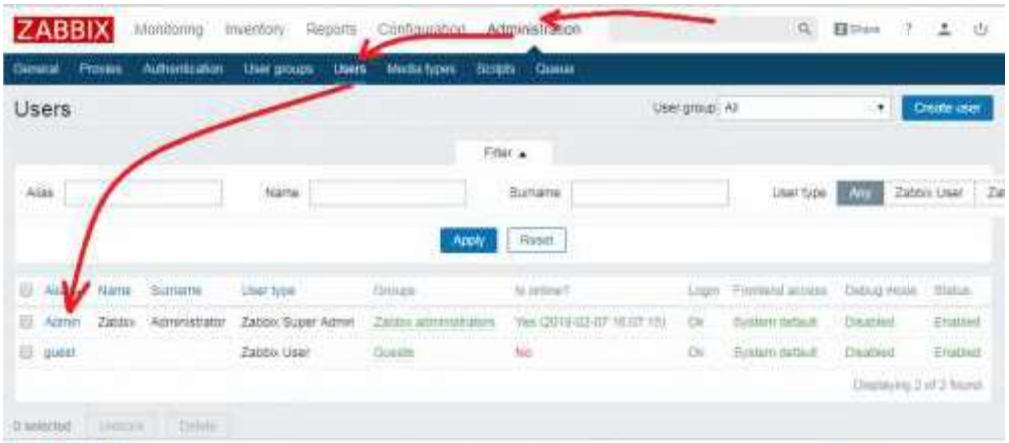
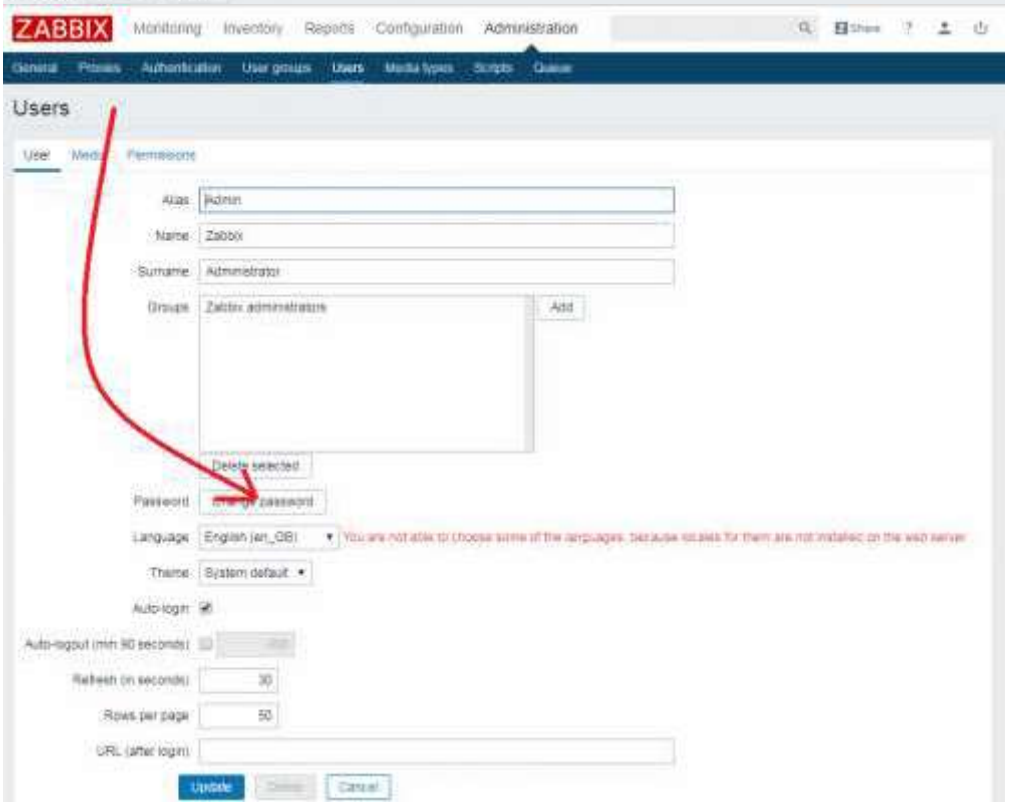
```
Hostname=vlbisysmon.vlbi
```

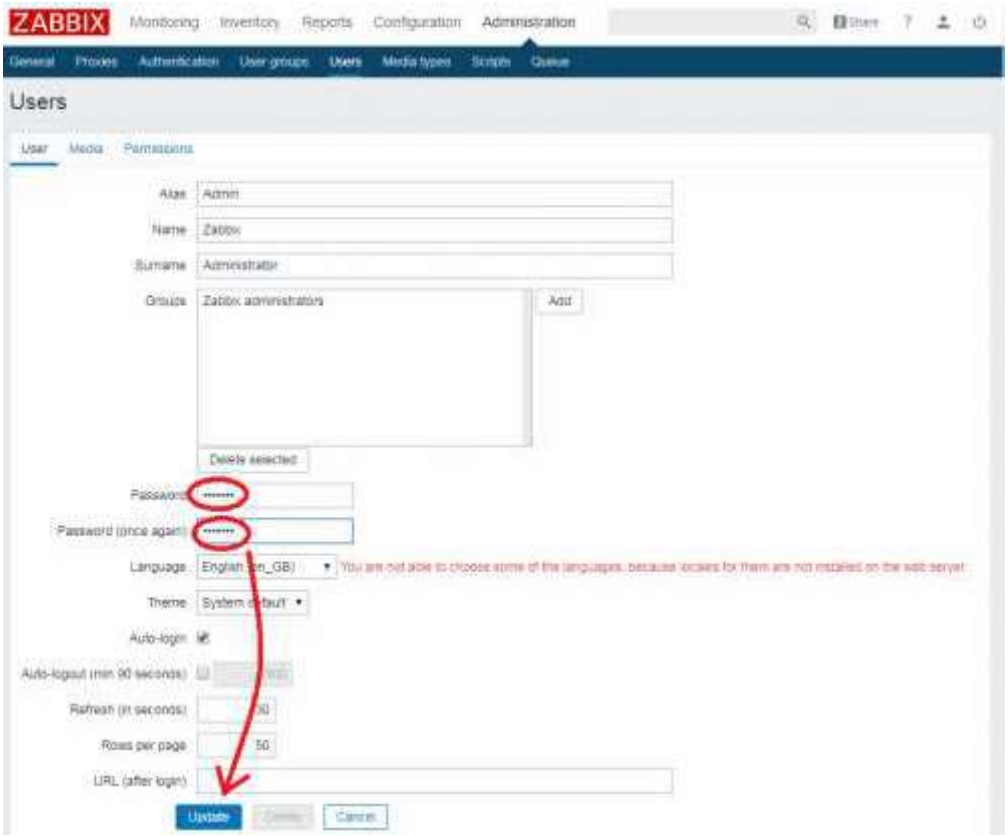
- Change hostname to “vlbisysmon.vlbi” also in the Web interface

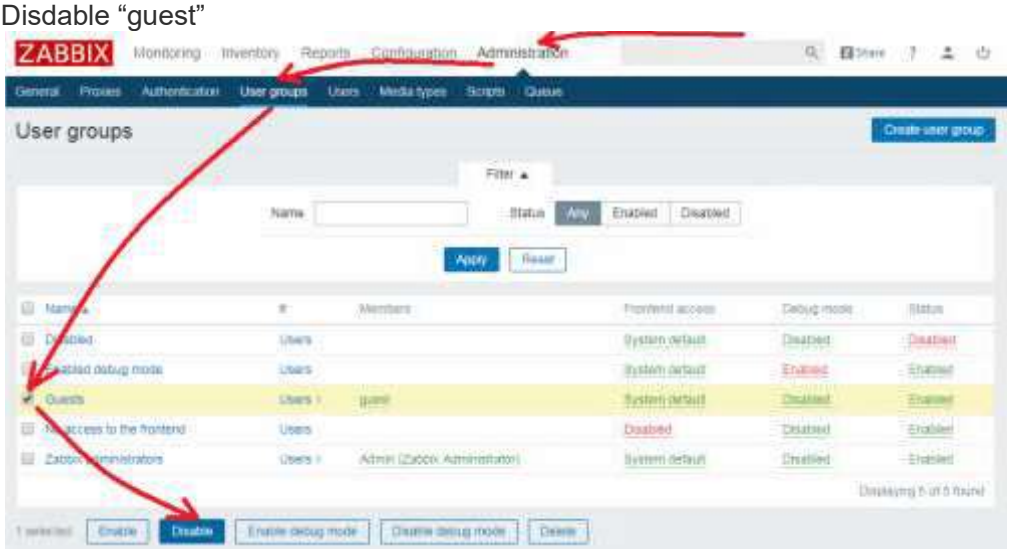




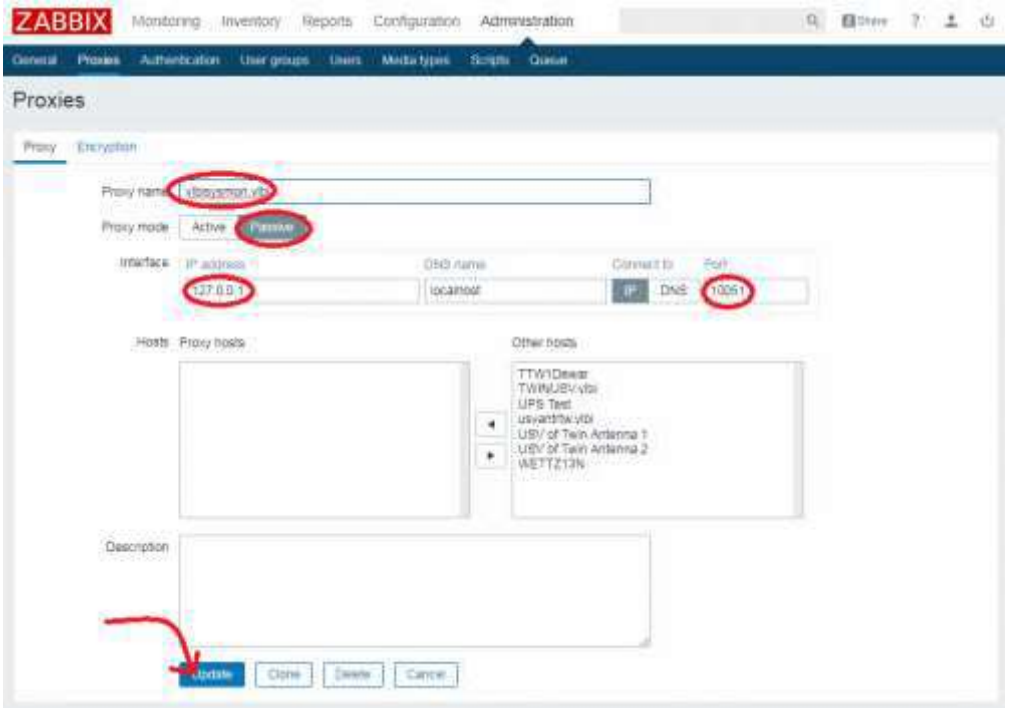
- Change password of user “Admin”

- 
ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
General
Proxies
Authentication
User groups
Users
Media types
Scripts
Queue
Users
User group All
Create user
Filter
Alias
Name
Surname
User type
Any
Zabbix User
Apply
Reset
Alias
Name
Surname
User type
Groups
Is online?
Login
Frontend access
Debug mode
Status
Admin
Zabbix
Administrator
Zabbix Super Admin
Zabbix administrators
Yes (2019-03-07 16:07:15)
Ok
System default
Disabled
Enabled
guest
Zabbix User
Guests
No
Ok
System default
Disabled
Enabled
Displaying 2 of 2 found
0 selected
Unblock
Delete
- 
ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
General
Proxies
Authentication
User groups
Users
Media types
Scripts
Queue
Users
User
Media
Permissions
Alias
Admin
Name
Zabbix
Surname
Administrator
Groups
Zabbix administrators
Add
Delete selected
Password
Change password
Language
English (en_GB)
You are not able to choose some of the languages, because locales for them are not installed on the web server
Theme
System default
Auto-login
Auto-logout (min 90 seconds)
Refresh (in seconds)
30
Rows per page
50
URL (after login)
Update
Delete
Cancel

- 

- Disdable “guest”
- 

- Create the proxy for the localhost in the Web interface
- 



- This proxy **must** now be used to access agents etc., because it is also used by other external Zabbix servers to forward data in distributed systems. **Attention: All data items which are just collected by a local Zabbix server and not by the proxy cannot be forwarded to external Zabbix servers outside of firewalls!!!**
- <u>**Possible error situations**</u>
  - Error log: *zabbix_agentd [8394]: Can't recreate Zabbix semaphores for IPC key 0x7a028449 Semaphore ID 196608. Operation not permitted.*
    - Remove the semaphore manually *ipcrm -S 0x7a028449*
  - Error log: *zabbix_server [56363]: cannot attach to existing shared memory: [13] Permission denied*
    - The program was started before using another user ⇒ always start programs with same user; maybe reboot to solve this problem
  - Data do not arrive at the zabbix_server
    - Stop the zabbix_server, zabbix_proxy and zabbix_agentd; start the zabbix_proxy; start the zabbix_server; start the zabbix_agentd

4.1.13 Create an HTTP file archive

- Create a directory in the web space of the already existing Apache server to store historic monitoring data there as files

```
mkdir /var/www/html/monitoring_archive
chown -R www-data:www-data /var/www/html/monitoring_archive
chmod -R 777 /var/www/html/monitoring_archive
```

- The structure of the archive can be individual but suggested is a folder structure in the following way: monitoring control point ID, year, month, individual day file, e.g.

```
TTW1Dewar
|-> 2017
|    |-> 01
|    |    |-> 20170101TTW1Dewar.txt
|    |    |-> 20170102TTW1Dewar.txt
|    |    |-> 20170103TTW1Dewar.txt
|    |    |-> 20170104TTW1Dewar.txt
```

```
▪      |    |    |-> ...
▪      |    |-> 02
▪      |    |-> 03
▪      |    |-> 04
▪      |    |-> ...
▪      |-> 2018
▪      |-> 2019
▪      |-> ...
▪      Meteo
▪      ...
▪
```

- Each program for the individual monitoring control point must take care on the individual structure itself.

4.1.14 Change HTTP to HTTPS

- See: http://lab4.org/wiki/Zabbix_Webftontend_%C3%BCber_HTTPS_verschluesseln
- **Create an SSL certificate (or buy one)**
  - Create a new directory for the new certificates and change to this

```
▪   mkdir -p /etc/ssl/certs/zabbix-server
▪   cd /etc/ssl/certs/zabbix-server
```

  - Create private key (you have to enter a pass phrase) and delete password

```
▪   openssl genrsa -des3 -out server.key 2048
▪   openssl rsa -in server.key -out server_nopw.key
```

  - Sign certificate

```
▪   openssl req -new -key server_nopw.key -out server.csr
▪   You are about to be asked to enter information that will be
    incorporated
▪   into your certificate request.
▪   What you are about to enter is what is called a Distinguished Name or a
    DN.
▪   There are quite a few fields but you can leave some blank
▪   For some fields there will be a default value,
▪   If you enter '.', the field will be left blank.
▪   -----
▪   Country Name (2 letter code) [AU]:DE
▪   State or Province Name (full name) [Some-State]:Bayern
▪   Locality Name (eg, city) []:Bad Koetzting
▪   Organization Name (eg, company) [Internet Widgits Pty Ltd]:TUM
▪   Organizational Unit Name (eg, section) []:FESG
▪   Common Name (e.g. server FQDN or YOUR name) []:192.168.208.236 (future
    IP or alias)
▪   Email Address []:alexander.neidhardt@mytum.de
▪
▪   Please enter the following 'extra' attributes
▪   to be sent with your certificate request
▪   A challenge password []:
▪   An optional company name []:
```

  - Create self-signed certificate

```
openssl x509 -req -days 999 -in server.csr -signkey server_nopw.key -
out server.crt
```

- **Activate SSL in Apache**
  - Activate SSL module

```
a2enmod ssl
```

  - Activate port 443 in the Apache configuration */etc/apache2/ports.conf* and add

```
<IfModule mod_ssl.c>
   Listen 443
</IfModule>
```

  - Change to directory /etc/apache2/site-enabled

```
cd /etc/apache2/site-enabled
```

  - Create symbolic link

```
ln -s ../site-available/default-ssl.conf default-ssl.conf
```

  - Create a virtual host by editing /etc/apache2/sites-available/default-ssl.conf; add something like this:

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
        ServerAdmin webmaster@localhost
        ServerName zabbix.example.com
        SSLEngine On
        SSLCertificateFile /etc/ssl/certs/zabbix-server/server.crt
        SSLCertificateKeyFile /etc/ssl/certs/zabbix-
server/server_nopw.key
        DocumentRoot /var/www/zabbix/
</VirtualHost>
</IfModule>
```

  - Deactivate HTTP

```
a2dissite 000-default
```

  - Change listen ports to avoid port 80 in /etc/apache2/ports.conf

```
#Listen 80
```

  - Restart Apache server

```
/etc/init.d/apache2 restart
```

4.1.15 Specific setup for the Wettzell vlbisysmon-PCs

- Wettzell VLBI-systems use the complete Wettzell suite of SysMon. It is available here: http://xsamba.wtz/svn/vlbi/trunk/code/vlbisysmon/
- All standard installation folders and files of Zabbix are deleted, because everything is contained in "*/home/oper/Software/vlbisysmon*"
- It contains the folders:
  - "**bin**": all executable binarys and script files after compilation and building
  - "**control**": all configuration files
  - "**main**": all main programs of the Wettzell SysMon suite

- “**make**”: a general Makefile to build and install the central vlbisysmon-PC with Zabbix-server, -proxy, and -agentd
- “**scripts**”: the central start script and all other scripts (usually as externals to the VLBI script folder)
- “**sensor_hardware**”: all code parts running on a separate sensor hardware
- “**sensor_proxies**”: all clients connecting to sensor servers to get data and copying them to SysMon
- “**sensor_servers**”: all sensor servers connecting to a hardware, reading data and offering them usually with an RPC interface
- “**sensors**”: all interface code files to communicate to the sensor hardware (client side); usually used by sensor servers to read data
- “**ssh_draft**”: SSH files, like private keys to automatically connect to other servers
- “**zabbix**”: Zabbix releases which are used on the system

- To build and install the VLBI SysMon system do the following steps
  - Create a folder “*Software*” in the home directory of user “*oper*”
  - Change into this folder

```
svn co http://xsamba.wtz/svn/vlbi/trunk/code/vlbisysmon/
cd ./vlbisysmon/make
```

  - Build and install the code

```
make
make install
```

  - Now everything is prepared for a start
- To start all servers run

```
/etc/init.d/vlbisysmon_server start
```

- To check if all servers are run enter

```
/etc/init.d/vlbisysmon_server check
```

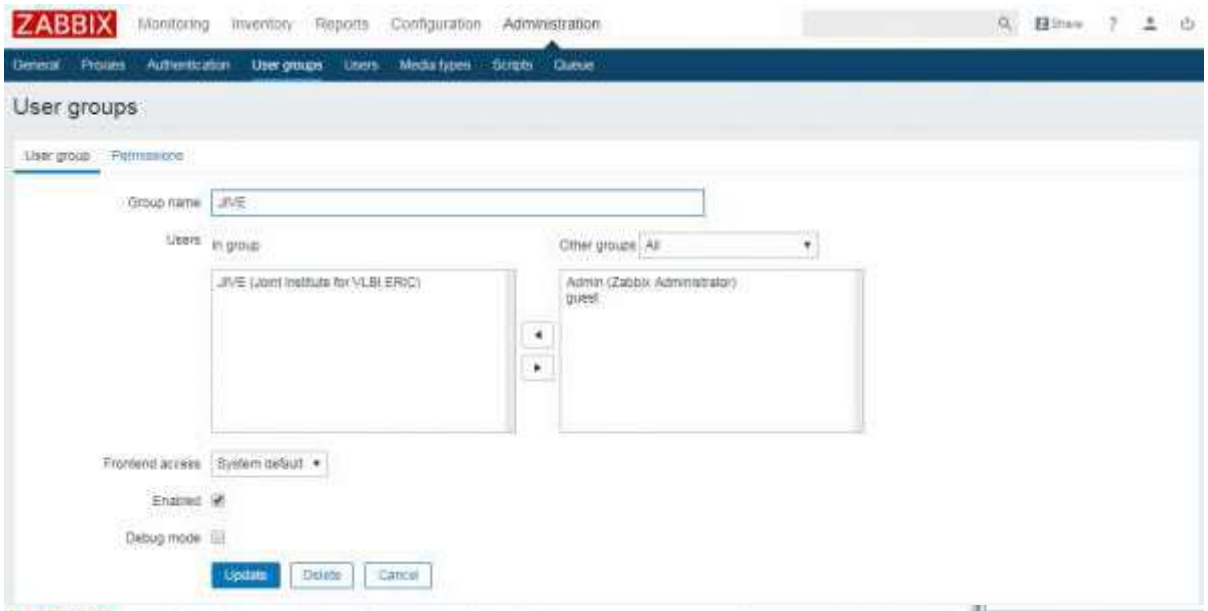
  - You should see something like this:

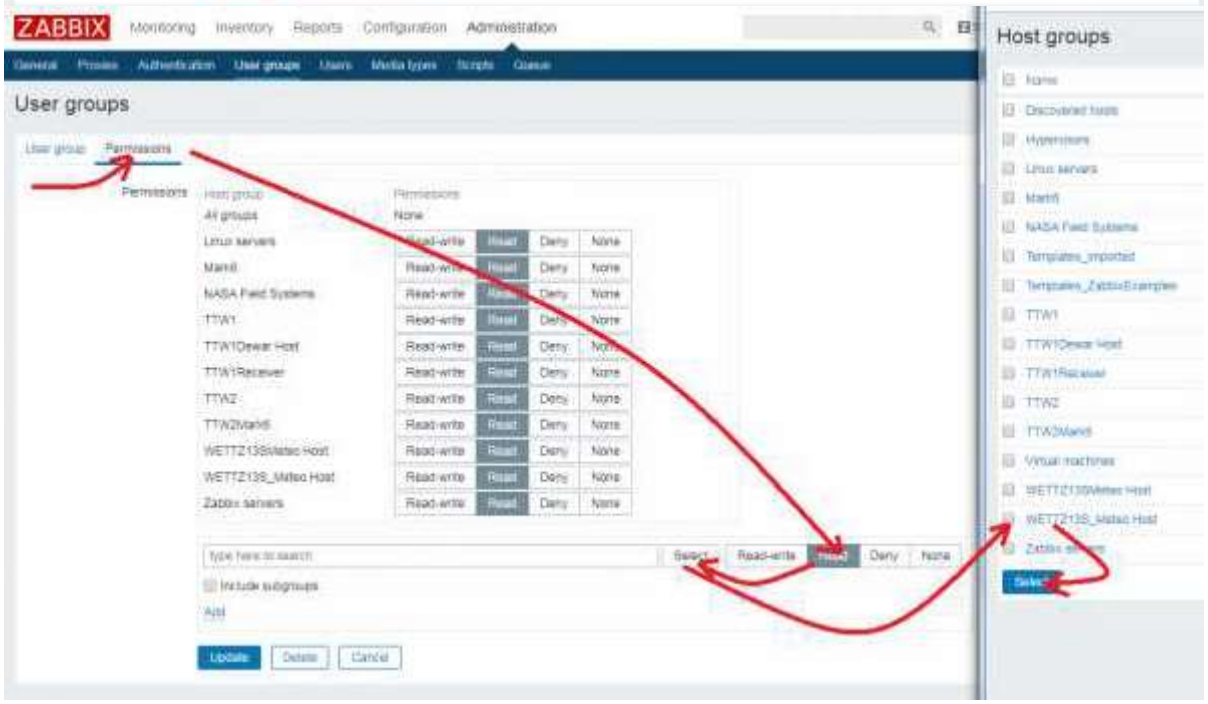
- To stop all servers enter

```
/etc/init.d/vlbisysmon_server stop
```

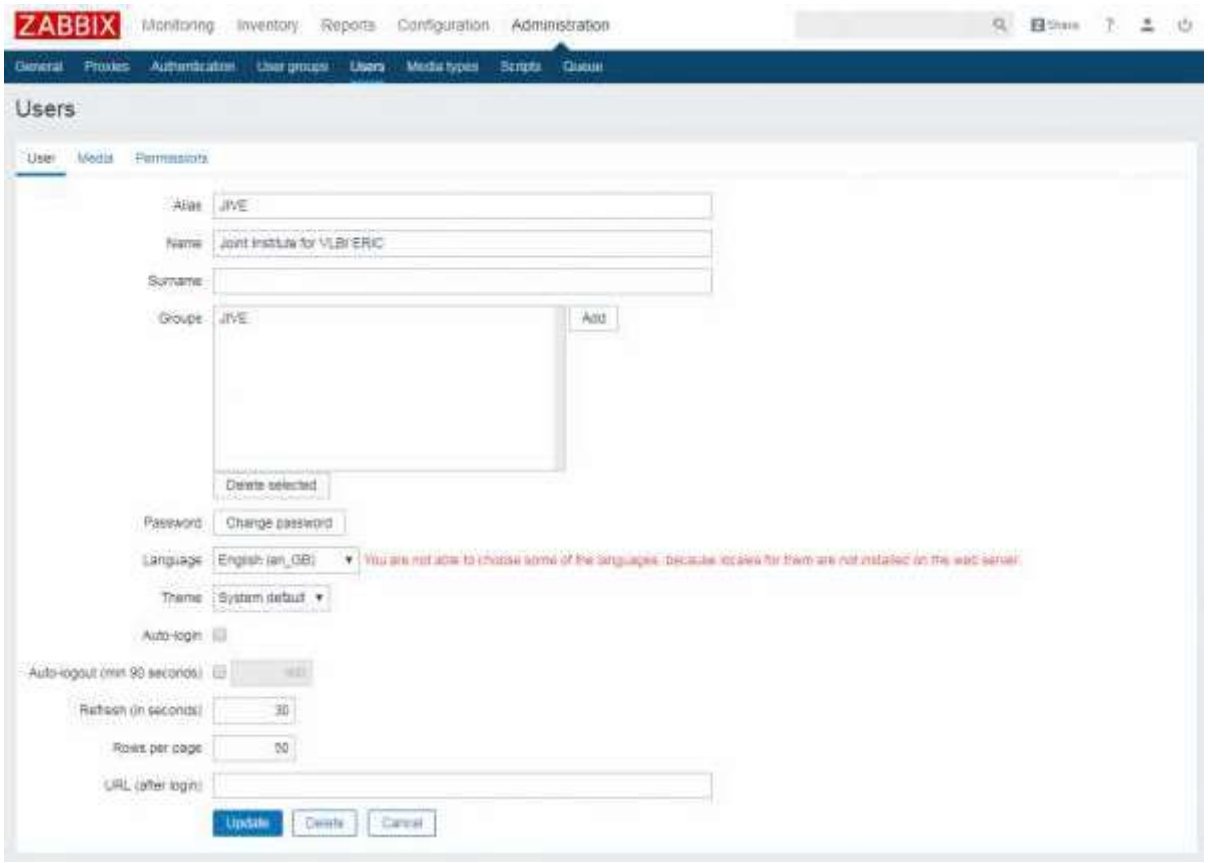
4.1.16 Create ZABBIX users for different purposes

- The following users are created on the centralized system monitoring machine
  - Admin (Zabbix Super User: administrator)
  - oper (Zabbix User: Wettzell operator)
  - JIVE (Zabbix User: Jumping JIVE EVN network manager)
  - maybe different other users one per site (Zabbix User: site operator)
- Before a user can be created, it is necessary to create a suitable user group with defined access rights (a user group and therefore a user has no rights at all after pure creation)
- A group can be created like this

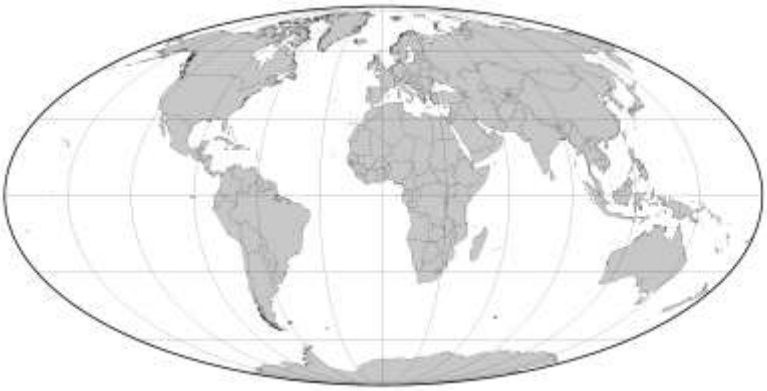
- 

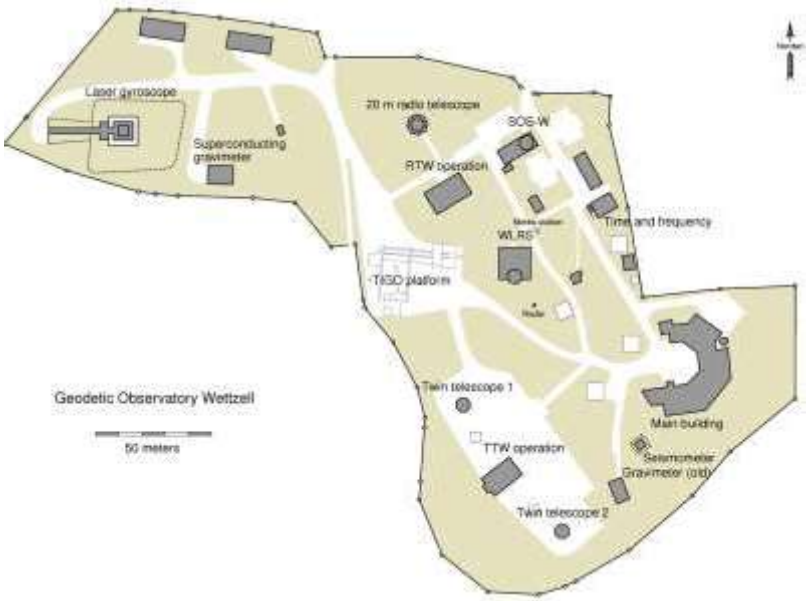
- 

- A user can then be created doing the following steps (defining a name, the group membership, password, etc.)
- 

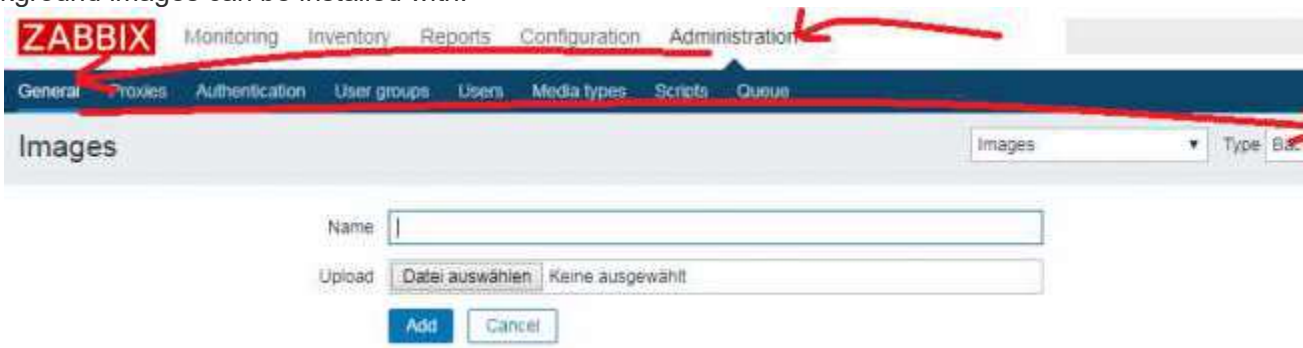
- 

4.1.17 Install additional images

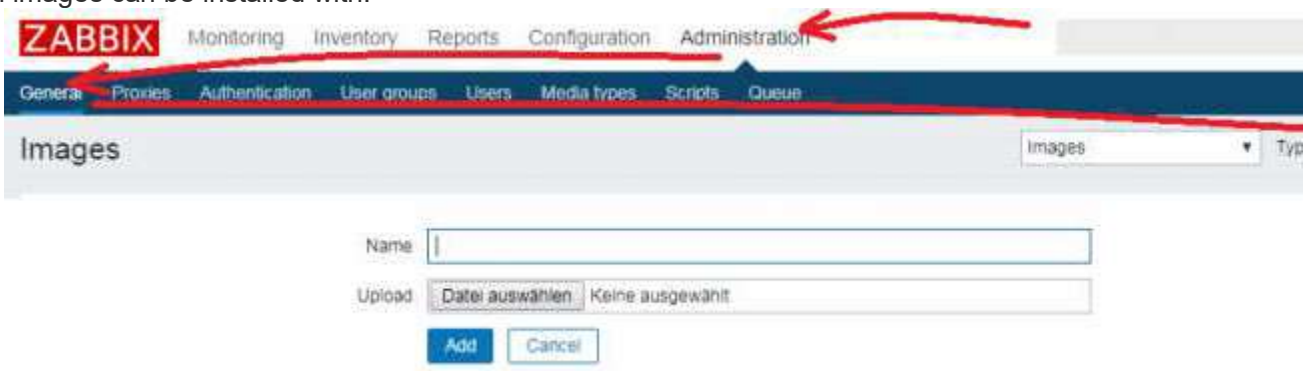
- The monitoring center uses additional background images:
- 
- world_sw_4000_2000.png
- world_sw_2000_1000.png
- world_sw_1000_500.png



-
  - stationmap_wettzell_2800_2100.png
  - stationmap_wettzell_1400_1050.png
- Background images can be installed with:



-
- The monitoring center uses additional icon images (icons always require the sizes 24, 48, 64, 96, 128 pixels):
  - 24 , 48 , 64 , 96 , 128 
- Icon images can be installed with:



-

4.2 Appendix: Installation and configuration of the monitoring of a NASA FS PC (with updates to D8.4)

The general data from the NASA FS PC are **operational or diagnostic data** which do not need to be saved for data VLBI analysis centers. Operational and diagnostic data are just managed in the Zabbix system and the data history is limited to one or two weeks (max. a month) to reduce data volume on the system monitoring PC.

4.2.1 Install a Zabbix agentd on the NASA FS PC

- The following description is just for NASA FS PCs which are in the same network as the Zabbix Server PC doing the monitoring.
- Download Zabbix sources as described in 0) SysMon Node VLBI
- Unpack sources e.g. into a folder /usr2/prog/Software/

```
tar -xzvf zabbix_3.2.4.orig.tar.gz
```

- Change into newly created directory, run the configuration utility and buidl the agent

```
cd /usr2/prog/Software/zabbix-3.2.4
./configure --enable-agent
make
```

- Make a safety copy of the original configuration file

```
cp /usr2/prog/Software/zabbix-3.2.4/config/zabbix_agentd.conf
/usr2/prog/Software/zabbix-3.2.4/config/zabbix_agentd.conf_orig
```

- Edit the configuration file and change the falues to the following (Hostname should be the name for the individual system which is also used later in the Zabbix front end; the IP addresses of the server must be those from the real Zabbix server PC)

```
LogFile=/tmp/zabbix_agentd.log
DebugLevel=3
Server=192.168.208.235
```

- Become root rights and do the following steps

```
su
cp /usr2/prog/Software/zabbix-3.2.4/conf/zabbix_agentd.conf
/usr/local/etc/zabbix_agentd.conf
cp /usr2/prog/Software/zabbix-3.2.4/src/zabbix_agent/zabbix_agentd
/usr/sbin/zabbix_agentd
groupadd zabbix
useradd -g zabbix zabbix
```

- Test the start of the Zabbix agent

```
/usr2/prog/Software/zabbix-3.2.4/src/zabbix_agent/zabbix_agentd -c
/usr2/prog/Software/zabbix-3.2.4/conf/zabbix_agentd.conf
```

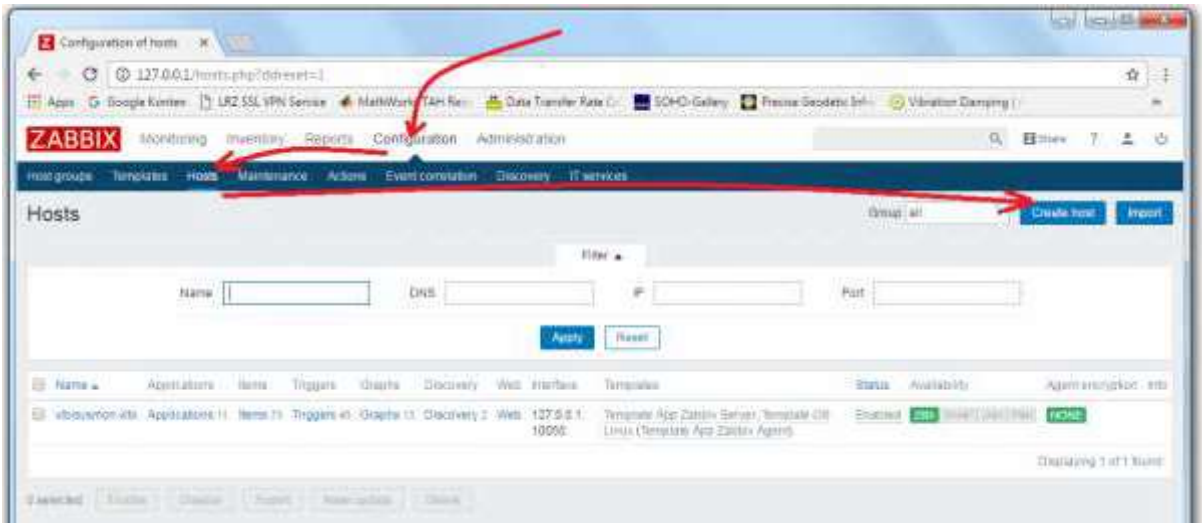
- Create a startscript in /etc/init.d/ e.g. with the name zabbix_agentd or include it to another start script. To start the zabbix_agentd it must contain this:

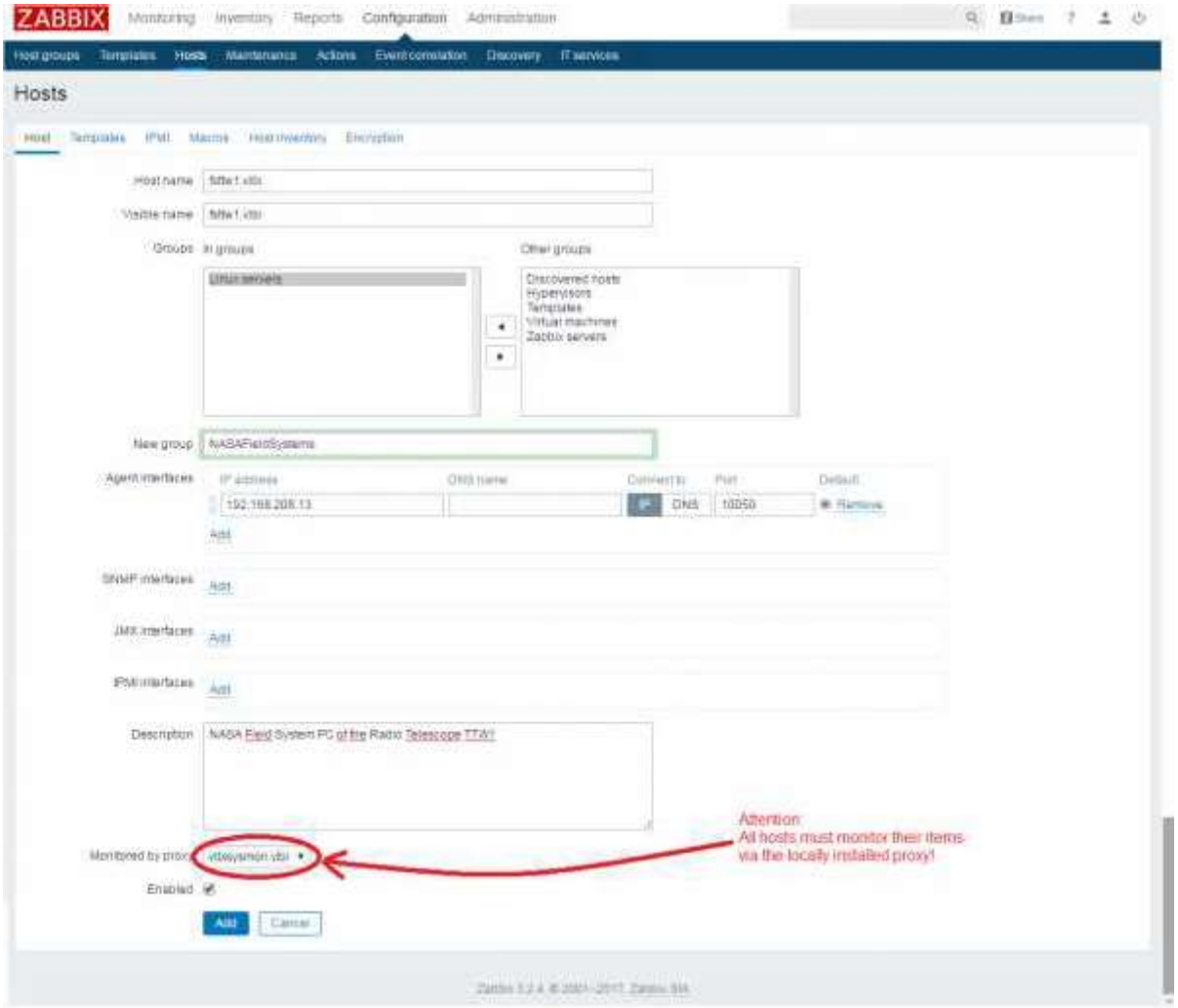
```
#!/bin/sh
### BEGIN INIT INFO
# Provides: zabbix_agentd
# Required-Start:
# Required-Stop:
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: Start zabbix_agent for PC monitoring
# Description: Start zabbix_agent for PC monitoring
### END INIT INFO

case $1 in
start)
    su daemon -c /usr/bin/zabbix_agentd
    ;;
stop)
    kill `cat /tmp/zabbix_agentd.pid`
    ;;
restart)
    $0 stop
    sleep 2
    $0 start
    ;;
*)
    echo "usage: $0 [start|stop|restart]"
    ;;
esac
```

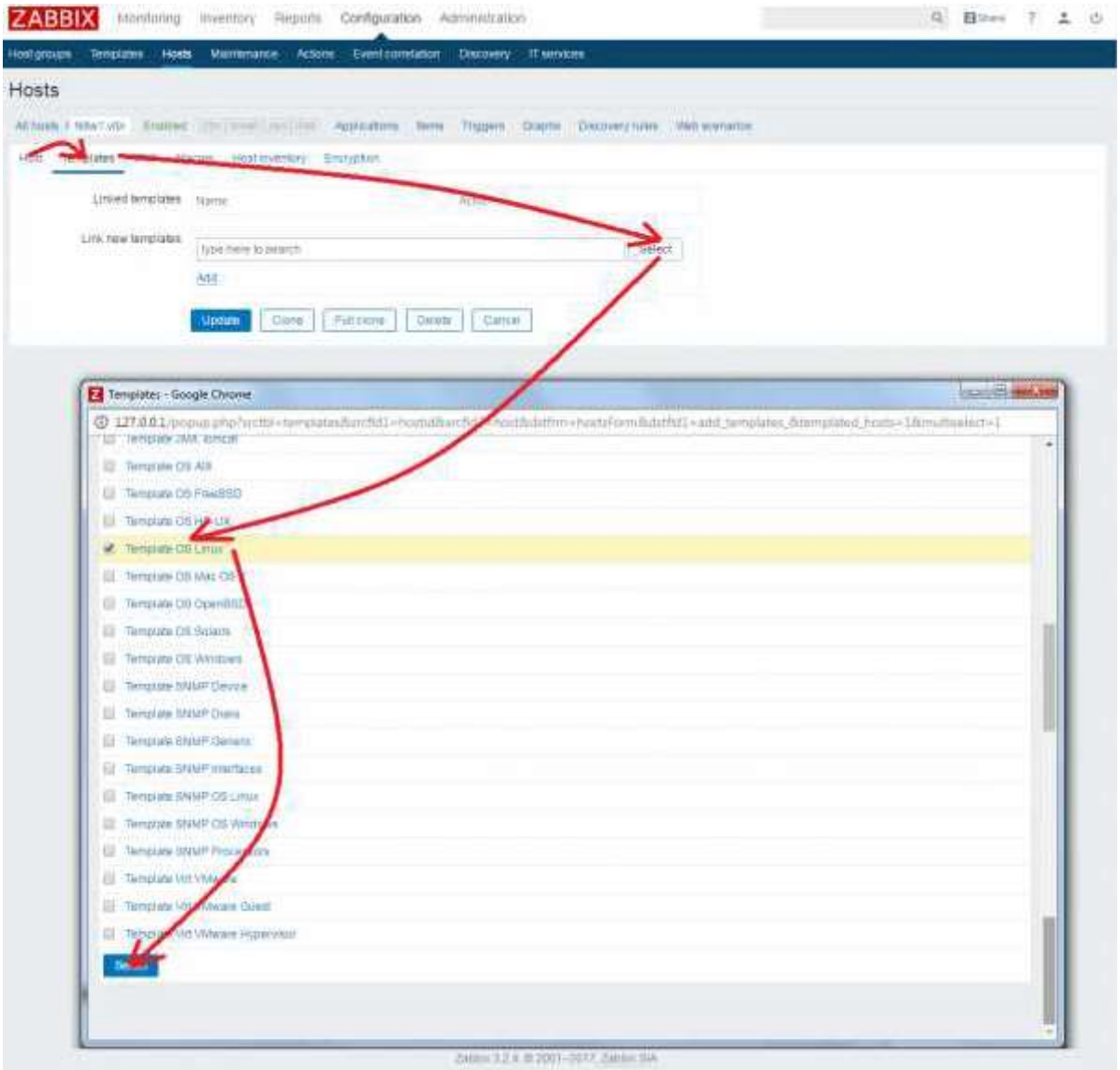
4.2.2 Activate monitoring on the Zabbix server

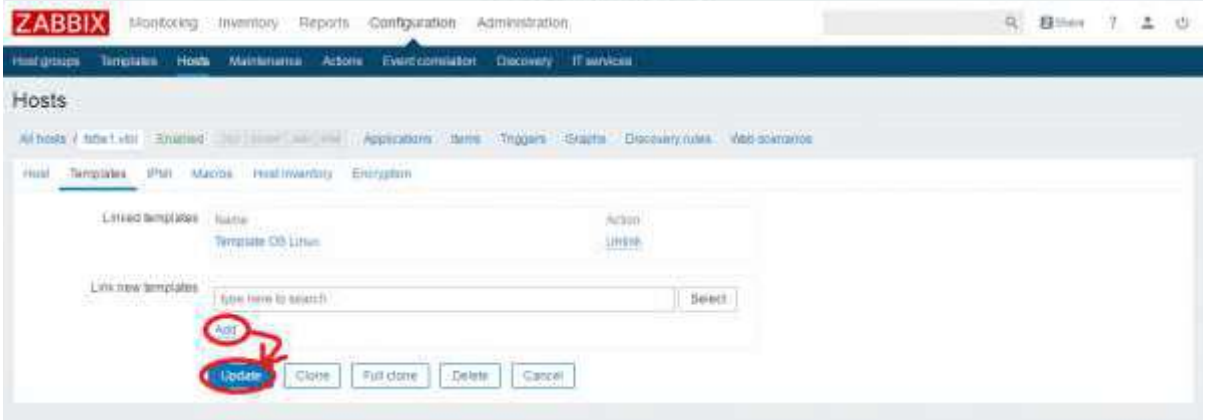
- Create new host





-
- Set templates

- 

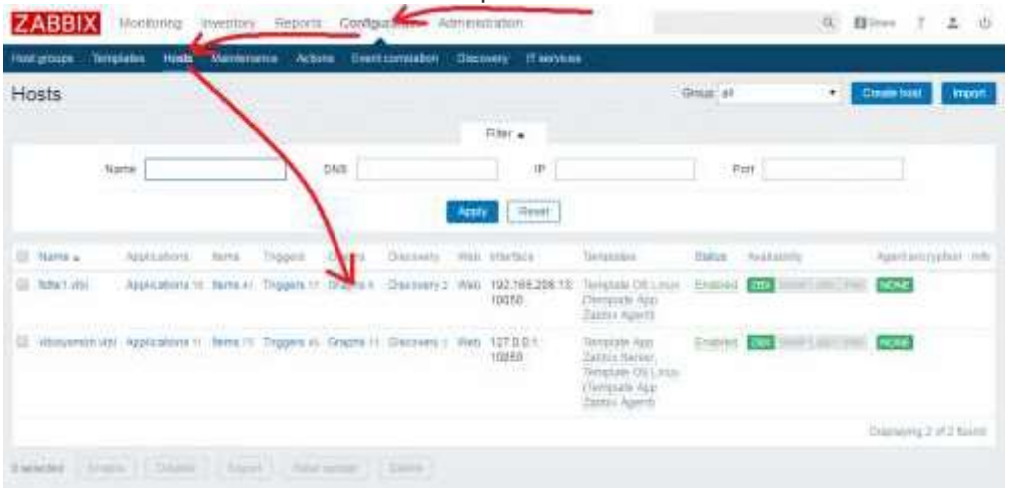
- 

- Wait a few minutes and the collected data should appear from the FSPC

4.2.3 Customize the data presentation/graph for the NASA FS PC needs

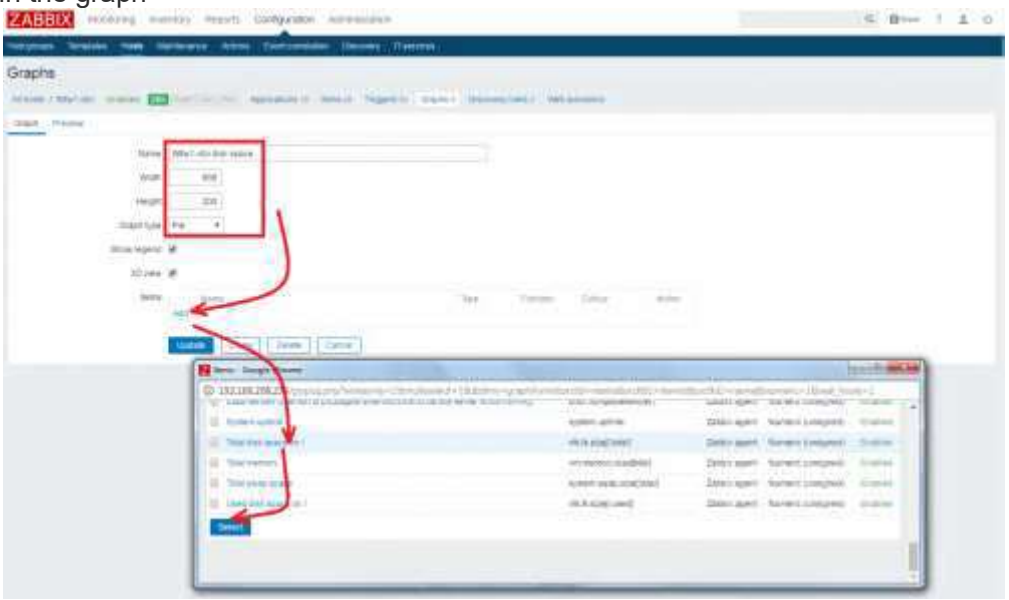
- Create new graph for disk space (which is an already existing item in the template for hosts and therefore already collected by the Zabbix agent)

- Select the host for the FS PC and “Graphs”

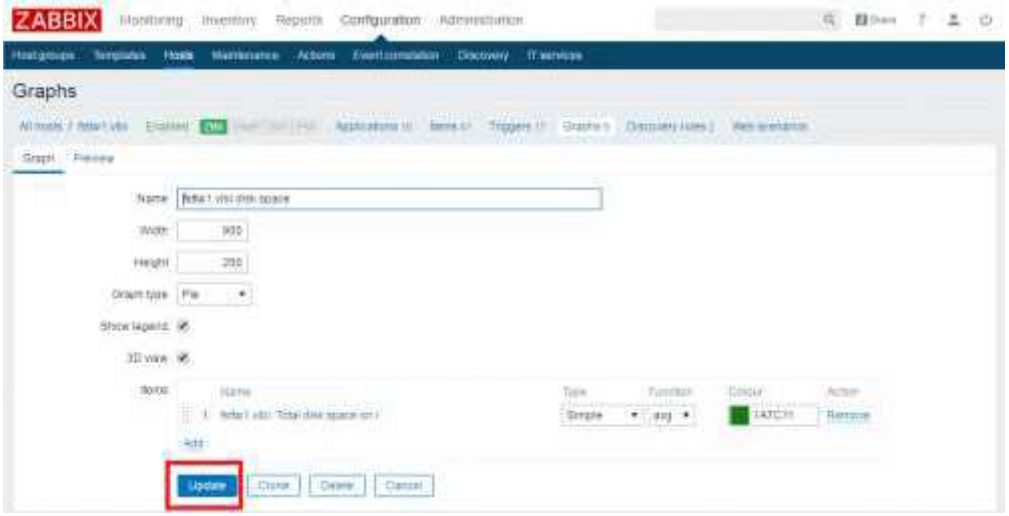


-
- Click on “Create graph”

-
- Enter a name, the graph sizes, the graph type (here “Pie” chart) and add the item monitored in the graph

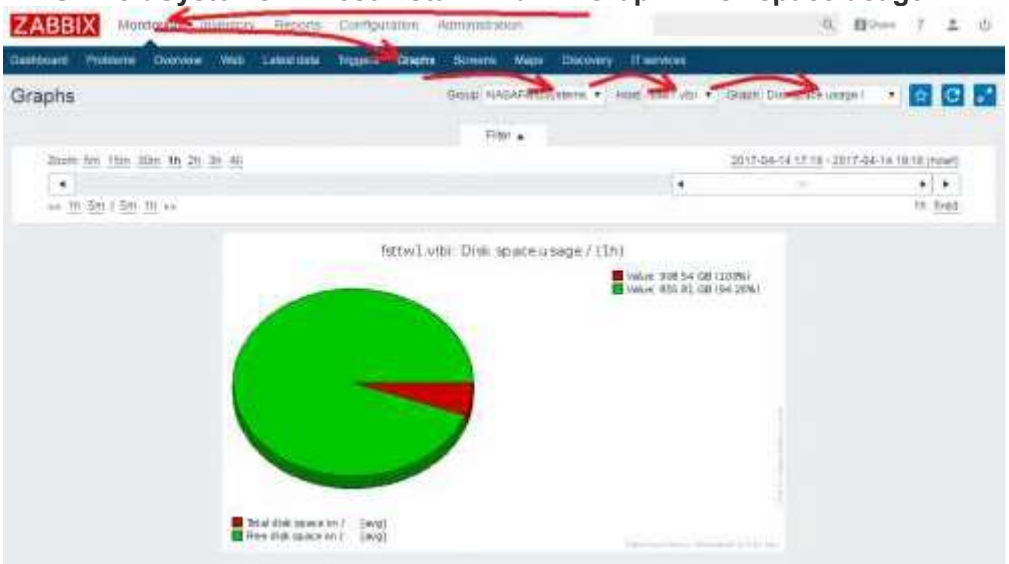


-
- Push “Update” to create new graph



-

- Check the output of the new graph by selecting **“Monitoring→Graphs→Group “NASAFieldSystems”→Host “fsttw1.vlbi”→ Graph “Disk space usage /“**



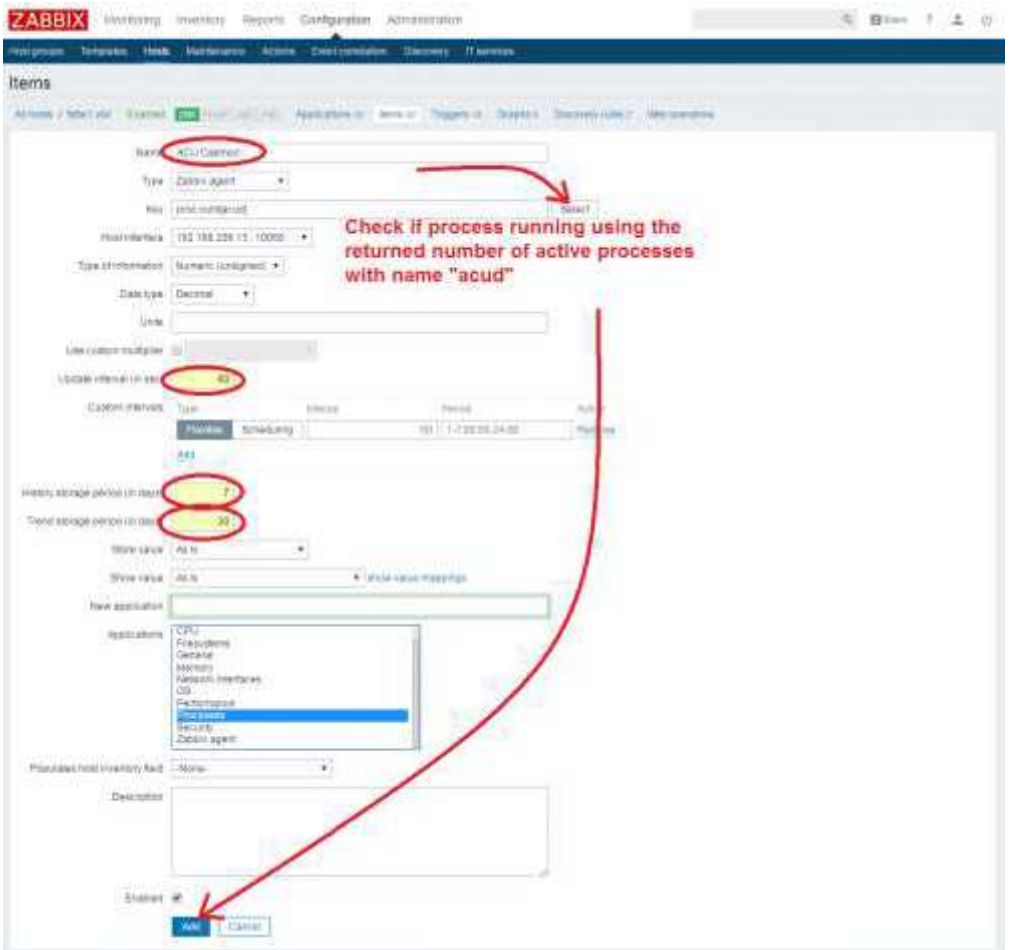
-
- This steps can now be done for all already collected monitoring data from the host “fsttw1.vlbi” or also for further computers from which data should be collected with a Zabbix agent host.

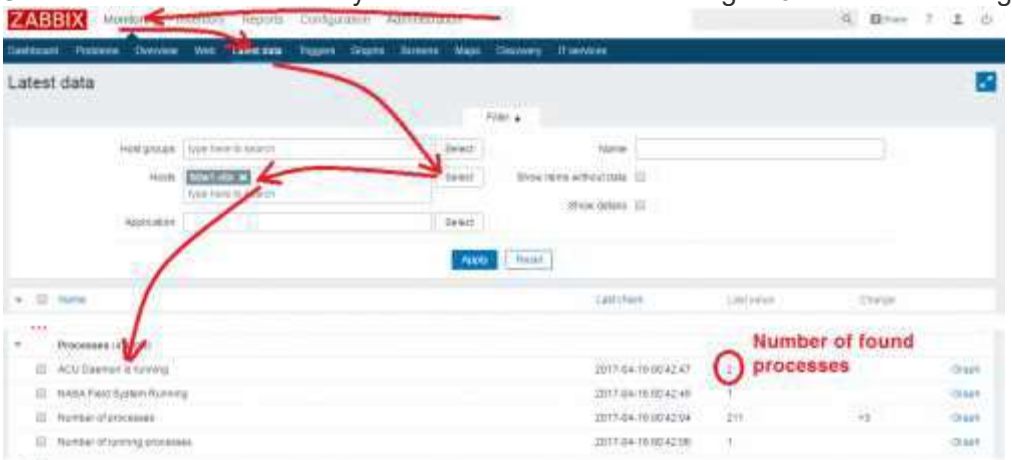
4.2.4 Add additional, individual monitoring items collected by Zabbix agent

- Create a new item (= monitoring sensor value) for the host “fsttw1.vlbi” by selecting “**Configuration→Hosts→“fsttw1.vlbi”:Items**”

-
- Click on the button “Create item”

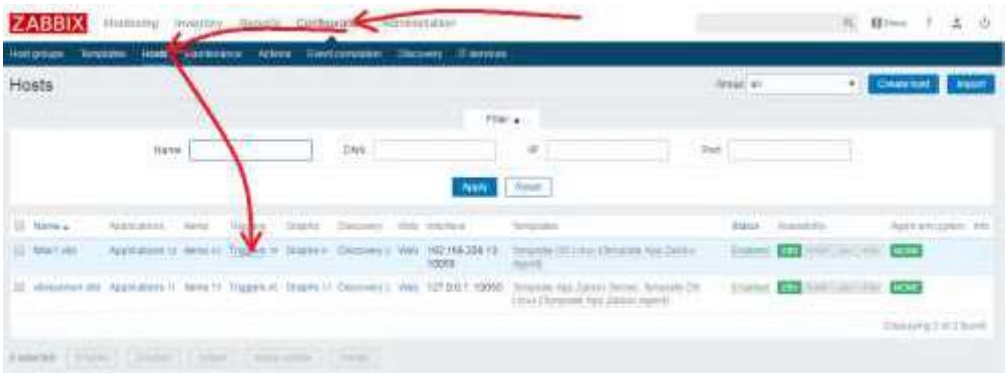
-
- One important item is the information if a process is running. Zabbix agent already supports such additional checks using specific keys. The key “proc.num” (number of process found for an individual name) is used for process checks. Enter a name for the item, e.g. to check if the Antenna Control Daemon (acud) is running, select the right key (e.g. proc.num(acud)), enter an update interval in seconds, enter the time for historic data storage and for trend storage, assign an application (e.g. Processes”), and push “Add” to create the new item for the host “fsttw1.vlbi”.

- 

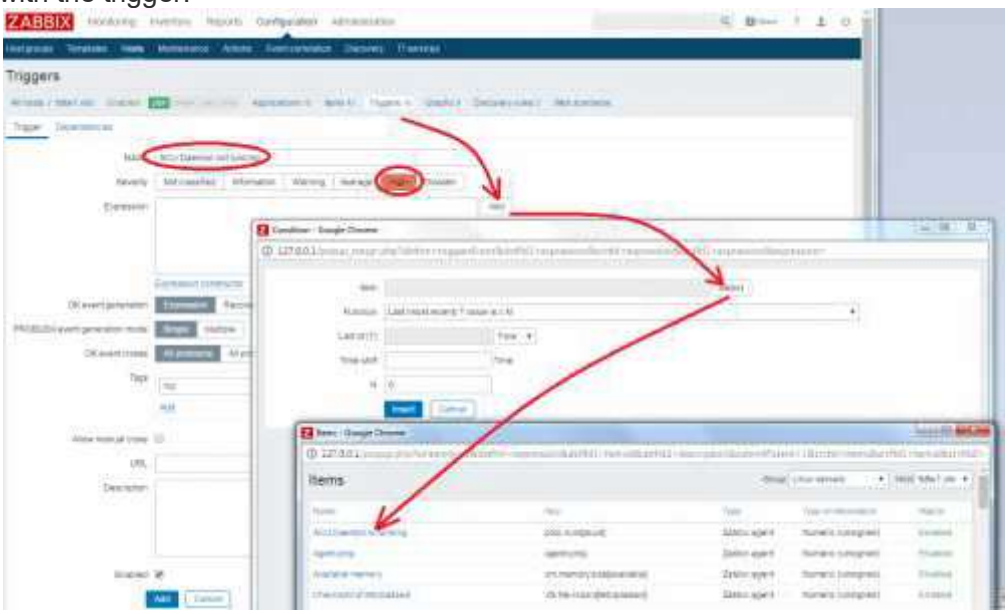
- Wait a minute and check if the data arrive for the new item using "**Monitoring→Select→ Hosts "fsttw1.vlbi"→Processes**" and check the column "Last value", which should show the number of processes found for the individual process name (for regular processes you should see 1 if it is running or 0 if it is not running; for "idl2rpc.pl" processes using the Wettzell Communication Middelware you should see 2 if it is running or 0 if it is not running).
- 

4.2.5 Add additional, individual trigger to detect alert levels

- Create a new trigger for alert levels for the host "fsttw1.vlbi" using "**Configuration→Hosts→"fsttw1.vlbi":Triggers**"

- 

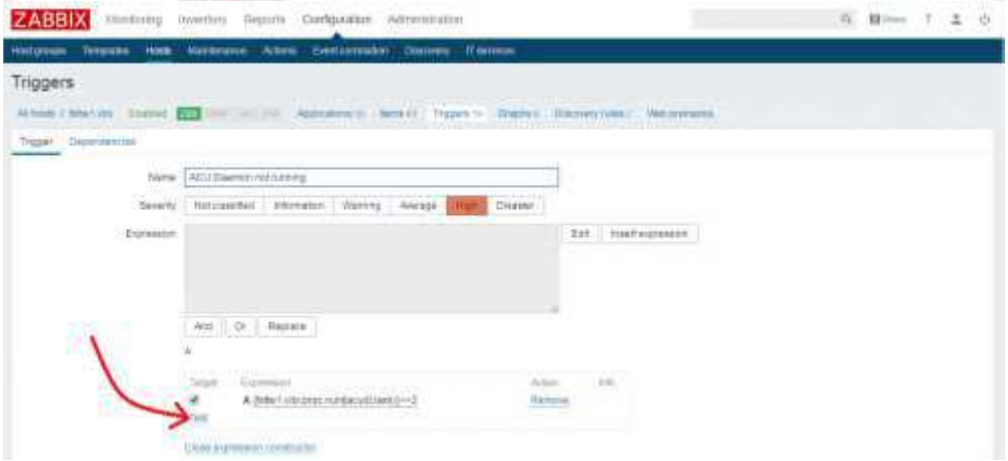
- Click the button “Create trigger”
- 

- Enter a name for the trigger (e.g. “ACU Daemon is not running”) and a severity (alarm level) and add an expression by pushing the button “Add”. Select the item which should be checked with the trigger.
- 

- Enter a function, which is the IF-statement to check which state will lead into an error state, e.g. all situations when not two idl2rpc.pl processes are found are errors here.
- 

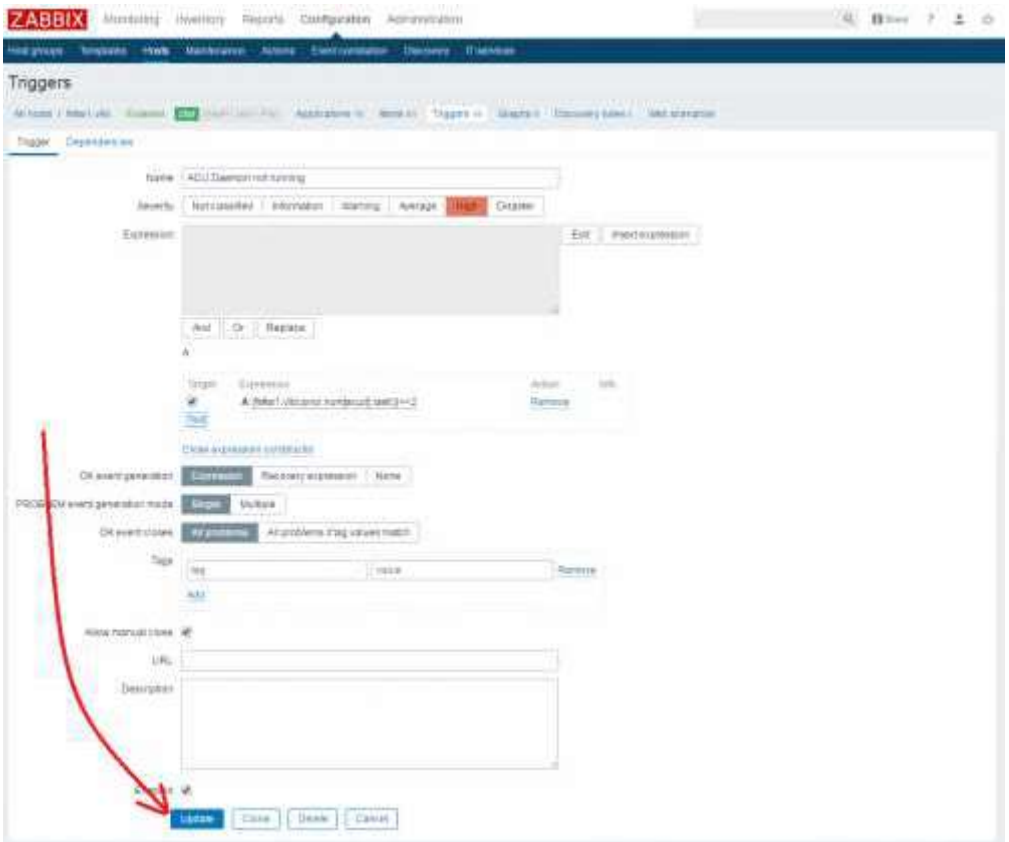
- To test the expression, it is necessary to open the Expression constructor and to click on “Test” for the activated expression.

- 

- 

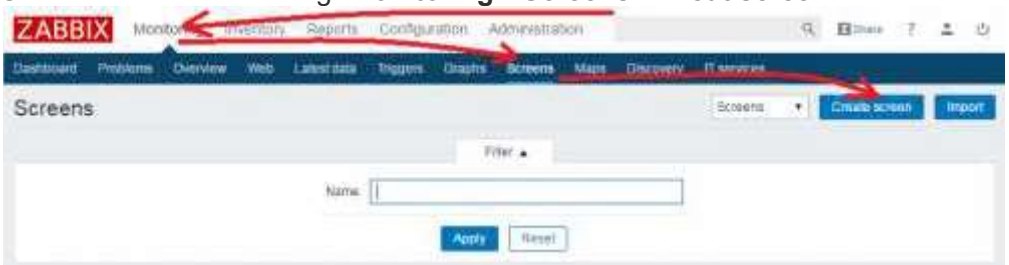
- The expression can then be tested against different values. The result of the expression will be shown in the “result” frame.
- 

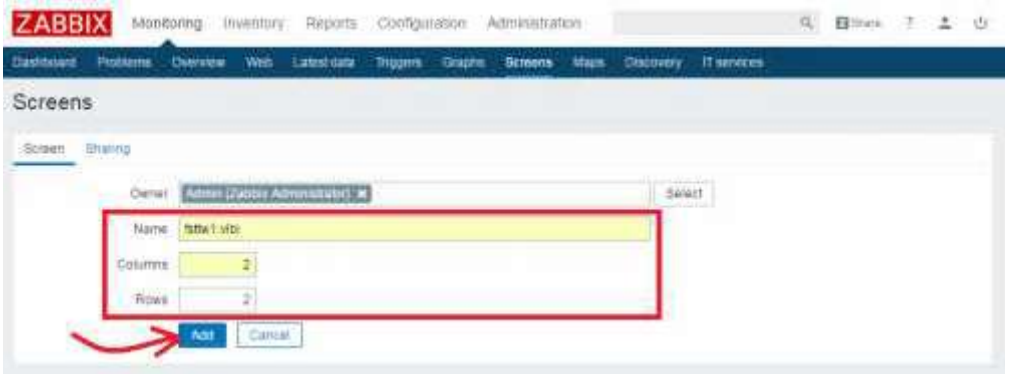
- Maybe also click “Allow manual close”, which means that an operator can reset the error state manually, and click “Update” to create the new trigger.

- 

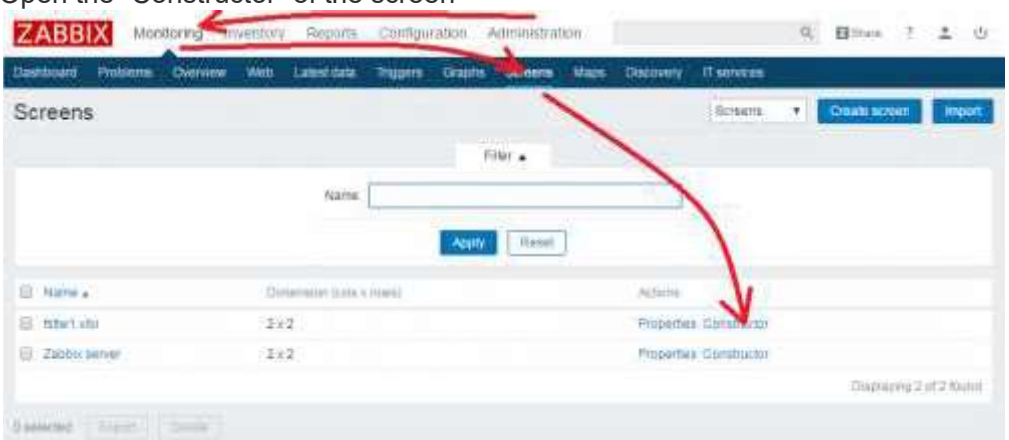
- The items are then tested and the host will show an error state if the trigger fires.

4.2.6 Create a screen to show all important information about the NASA FS PC

- Individual screens can be used to show system states and graphs. The most important states for the NASA FS PC are e.g. disk space, CPU load, maybe memory, network load.
- Create a new screen using "**Monitoring→Screens→Creat screen**"
- 

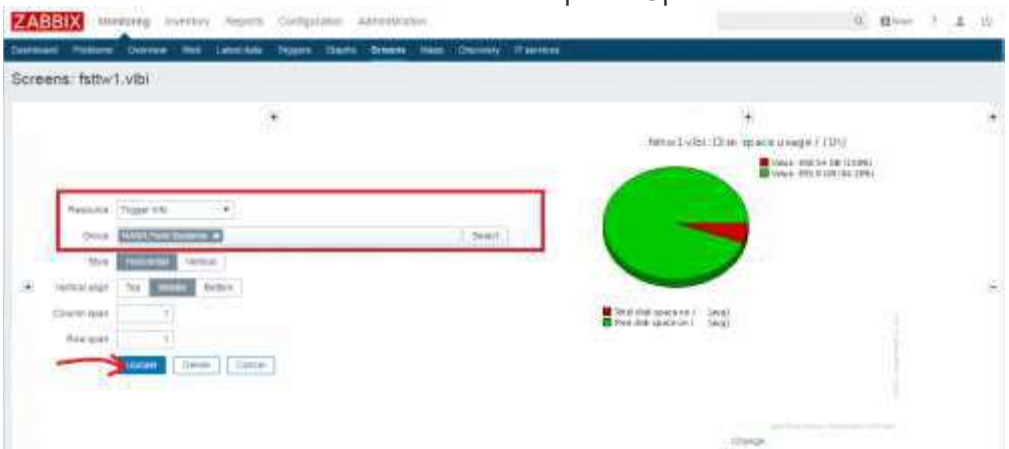
- Define a name for the screen and the dimensions as number of rows and columns and push "Add".
- 

- Open the “Constructor” of the screen

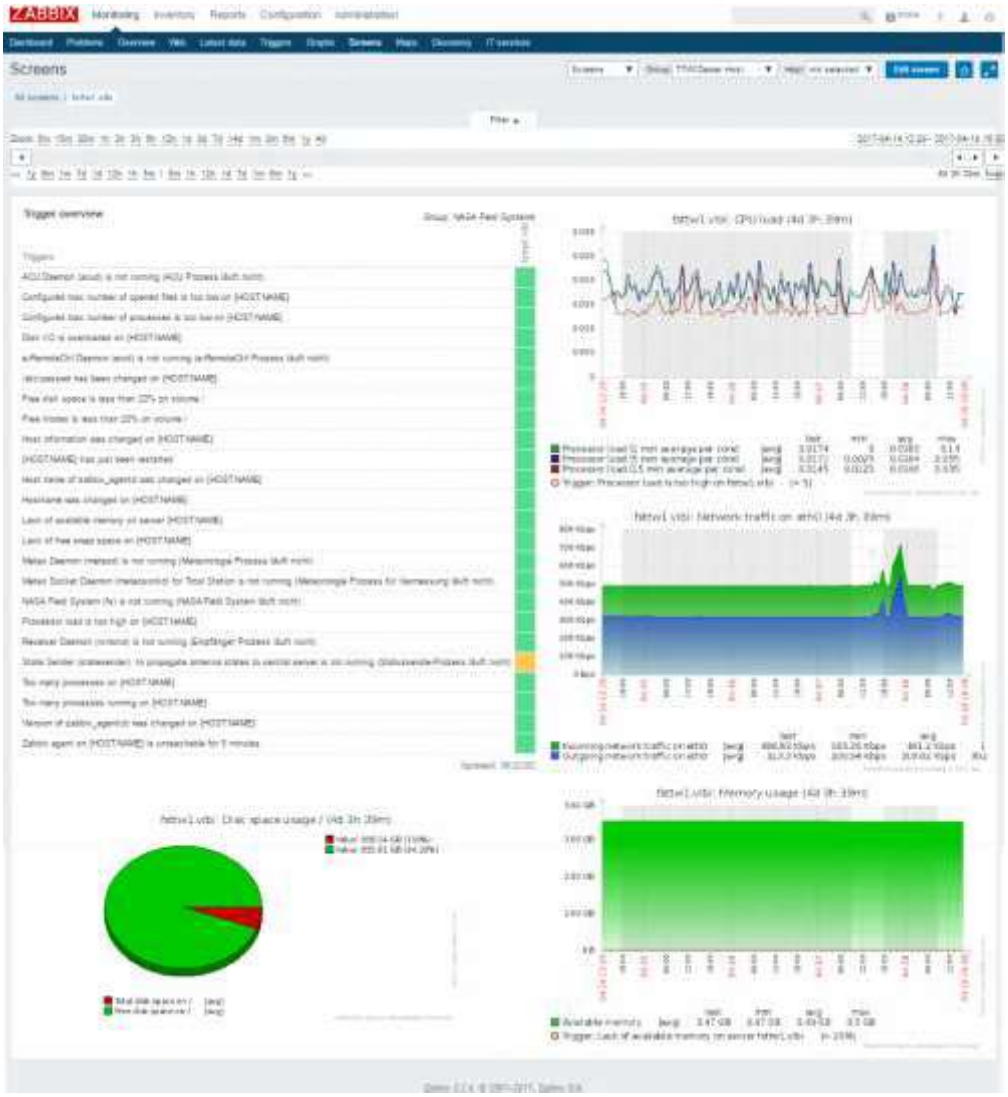


-
- Push the “Change” link for each of the individual fields on the screen.

-
- Select what should be shown in this field and push “Update” or “Add”.



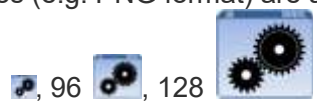
-
- If all fields are filled with parameters, the screen can look like this:

- 

4.2.7 Create an overview system map for the NASA FS PC needs

Maps are used to show the logic structure of system processes. They are a hierarchical structure for VLBI systems. The complete antenna system (e.g. TTW1) is one upper layer. It is split into sublayers of maps according to the individual parts, e.g. Control, Antenna, Infrastructure, IT, and so on. The structure can be individually.

4.2.7.1.1 7.1) Create own icons for individual processes

- To adapt to individual processes, individual icons for elements in the map can be created an load to Zabbix. New icons can be created with MS Powerpoint Sample icon in MS Powerpoint or graphic tools. It is important that different images (e.g. PNG format) are always created for the following width sizes in pixel: 24  , 48 , 64 , 96 , 128 pixels. These different sizes are used to increase a highlighted icon if an alert level is fired.
- A new icon can be added using "**Administration→general→Images**"

- 

- Push the button “Create icon” to install a new icon set.
- 

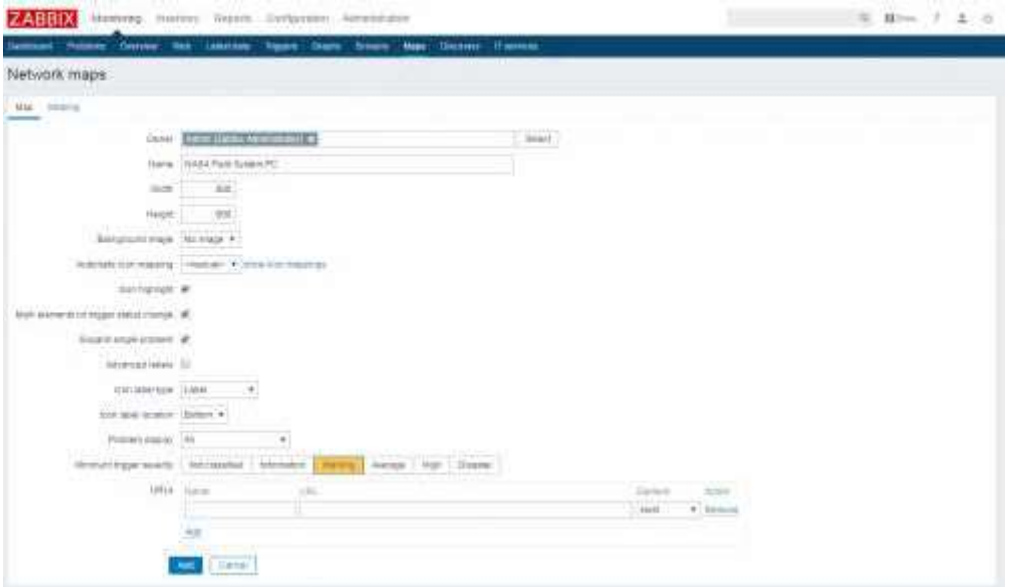
- Select the icon image file in the upload dialog and give a name for the icon, e.g. the icon type in combination with the size.
- 

- Do this step for all individual icons

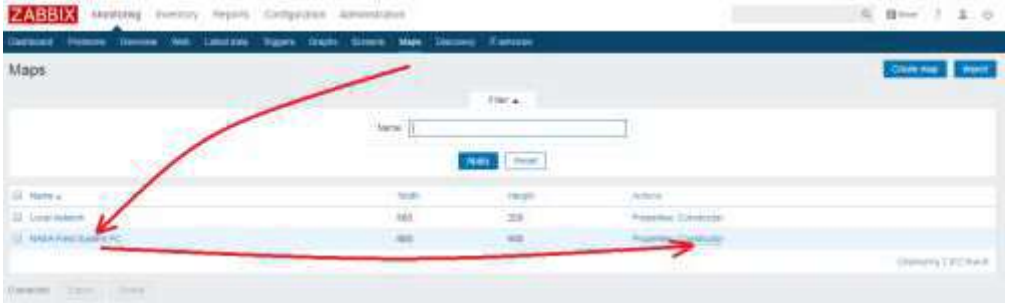
4.2.7.1.2 7.2) Create a new map for the NASA field system PC

- Create a new map for the NASA Field System PC using “**Monitoring→Maps→Create map**”
- 

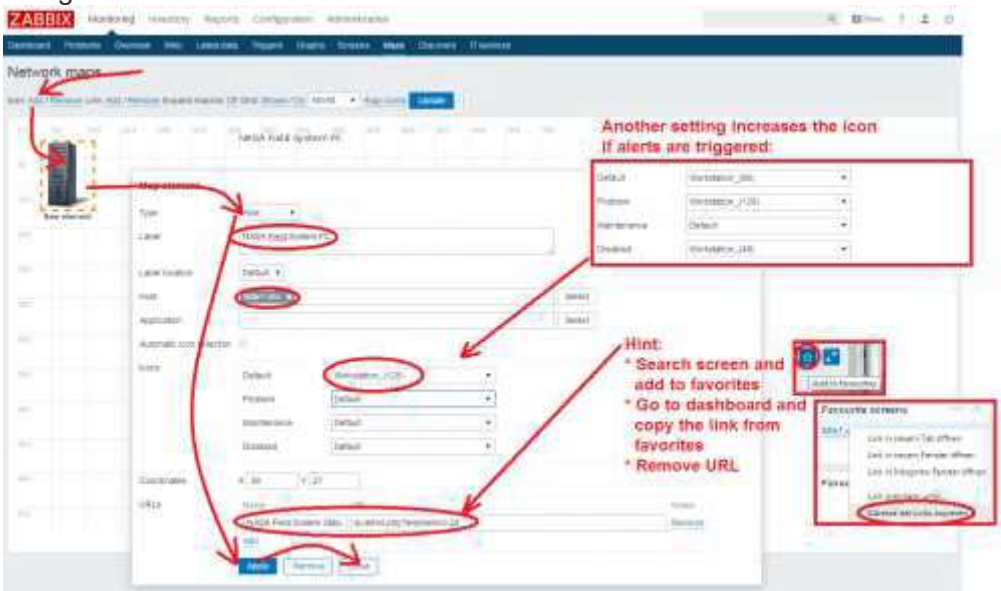
- Enter an owner. Also other users can be defined here to be owner of the map. Define a name for the map, e.g. “NASA Field System PC”, the dimensions of the map and check the checkboxes. Minimum trigger severity should be “Warning”, so that warnings and higher alerts are highlighted. Finally, press the button “Add” to create the map.



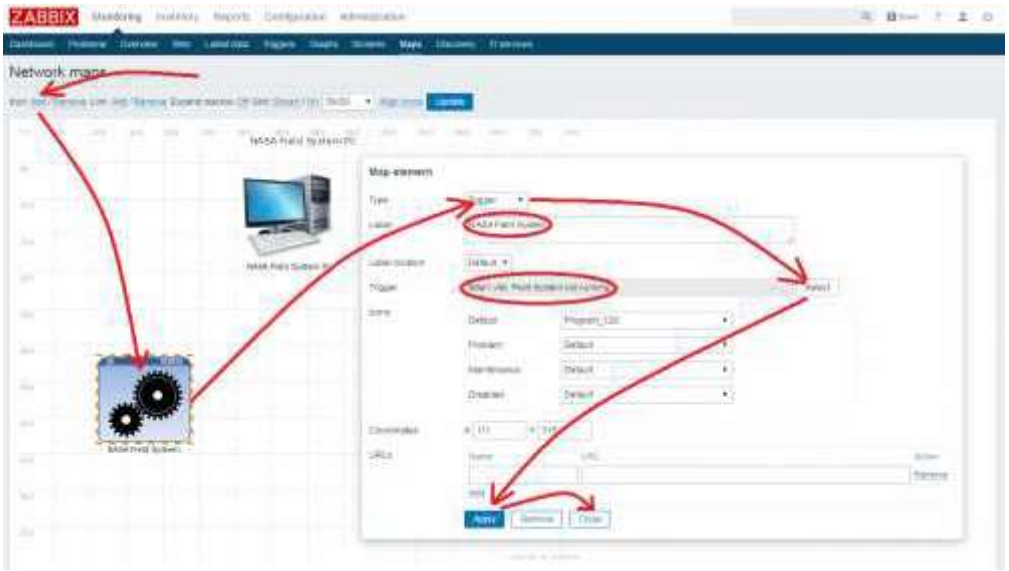
- Construct the new map to set icons and network connections clicking on the “Constructor” of the the newly created map.

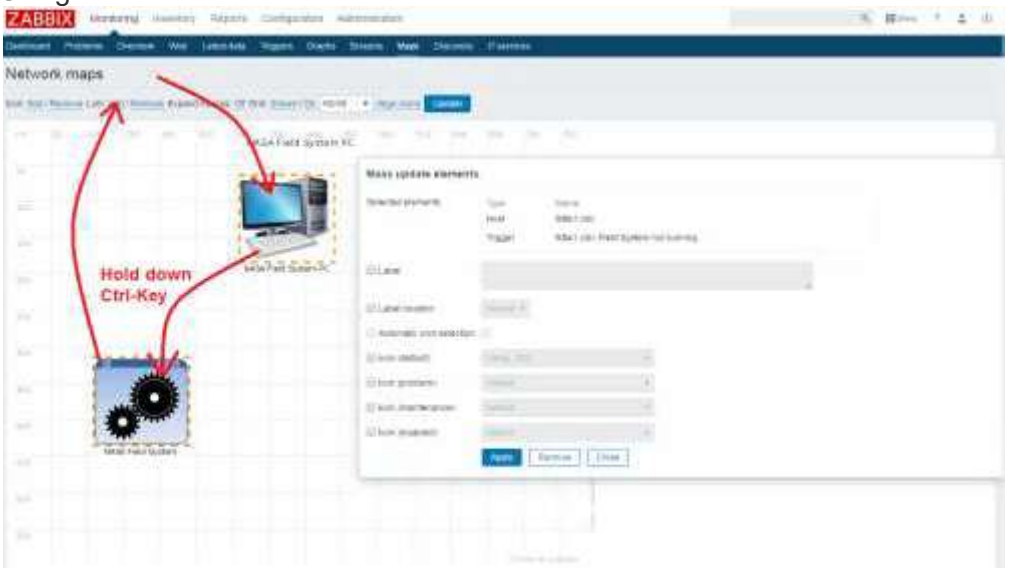


- Add a new icon and pull it to a position on the map. Click on the new icon and fill out the form which opens. Select the type (e.g. Host or Trigger or …), set the label, asign a host, define the different sizes of the icons for different alert situations and add an individual screen link to the URLs (use the internal references shown in the image below). “Apply” and “Close” the dialog.

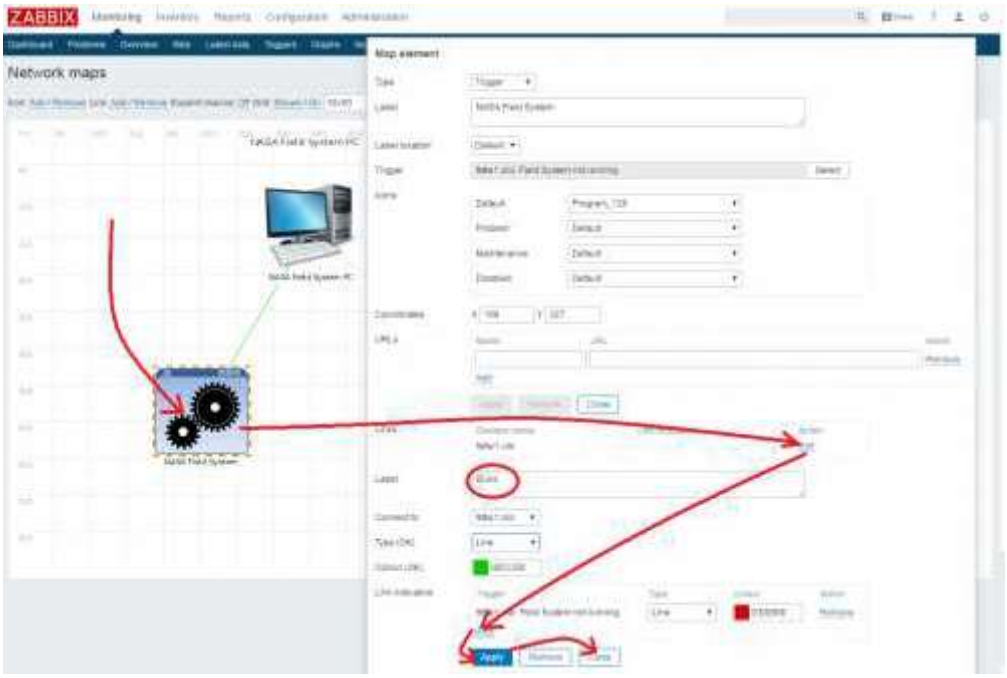


- Add further icons, e.g. for triggers

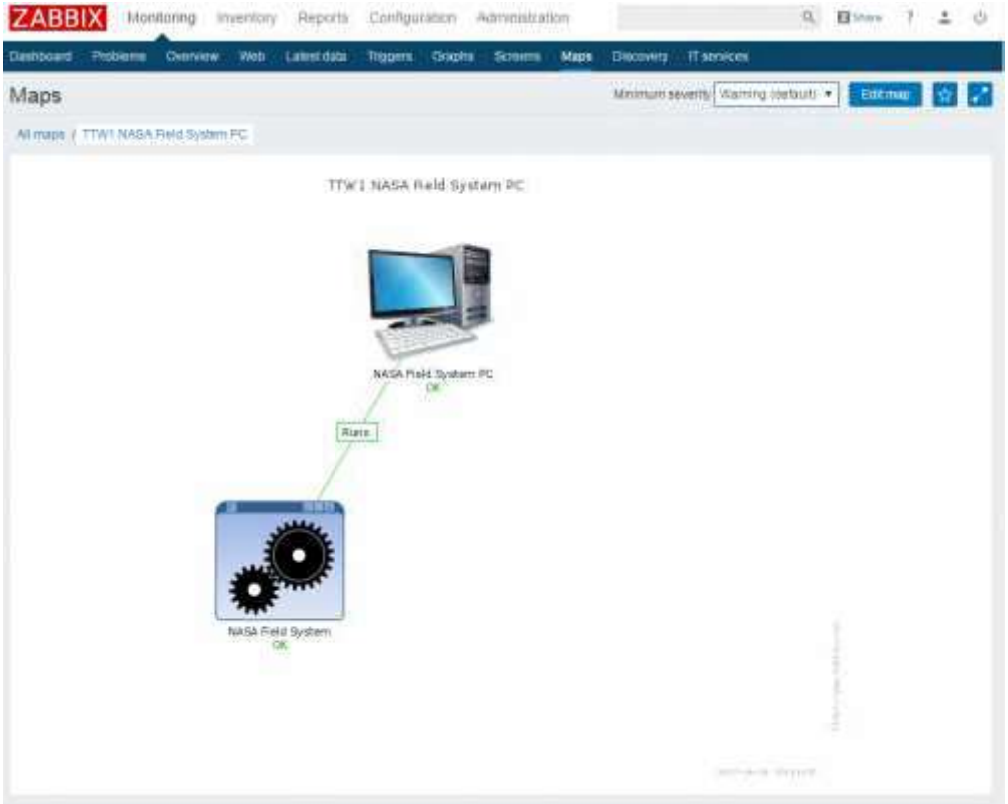
- 

- Link the individual icons. Click on one icon, hold doan the Ctrl-key and click on another icon. Using “Add” of a “Link” to connect both icons.
- 

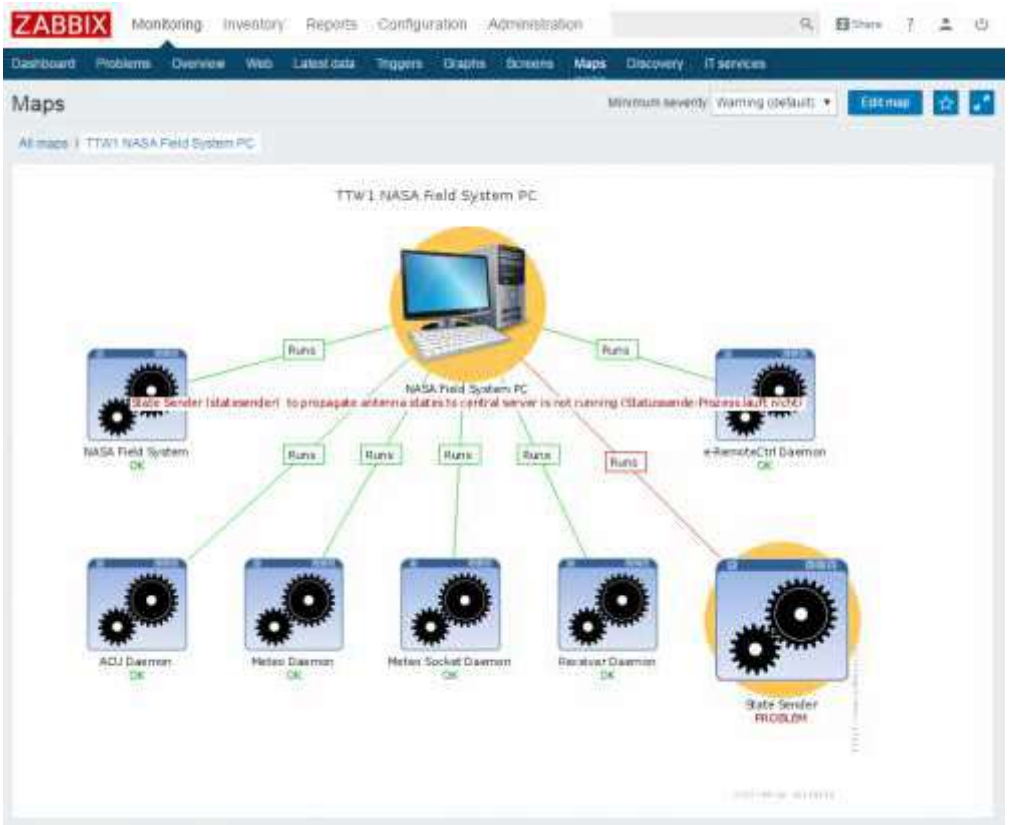
- Label the newly created link and connect it to a trigger, so that the line will be colored when the trigger fires.



- The finished link should look like this:



- If several system parameters are connected and set, the map with a warning severity looks like this:


ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
Dashboard
Problems
Overview
Web
Latest data
Triggers
Graphs
Screens
Maps
Discovery
IT services
Maps
Minimum severity
Warning (default)
Edit map
All maps / TTW1 NASA Field System PC
TTW1 NASA Field System PC
Runs
NASA Field System PC
NASA Field System
OK
e-RemoteCtrl Daemon
OK
ACU Daemon
OK
Meteo Daemon
OK
Meteo Socket Daemon
OK
Receiver Daemon
OK
State Sender
PROBLEM

4.3 Appendix: Installation and configuration of the monitoring of a Mark6 data recorder

In general, data from the Mark6 data recorder are **operational or diagnostic data** which do not need to be saved for data analysis at the VLBI analysis centers. Operational and diagnostic data are just managed in the Zabbix system and the data history is limited to one or two weeks (max. one or two month) to reduce data volume on the system monitoring PC.

4.3.1 Install a Zabbix agentd on the Mark6

- The following description is just for Mark6 systems which are in the same network as the Zabbix Server PC doing the monitoring.

4.3.2 Simplified installation using the Wettzell Mk6 station code (suggested way)

- Wettzell uses an already prepared installation package on a subversion repository which includes all necessary elements. All installations should be made on the Mark6 using the folder
  - */home/oper/Software/*
- As member of the Wettzell staff you have access to the latest version which can be found here:
  - http://xsamba.wtz/svn/vlbi/trunk/code/mk6stationcode/
  - Change into the installation folder */home/oper/Software/* and get the version with:

```
svn co http://xsamba.wtz/svn/vlbi/trunk/code/mk6stationcode/
```

- Non-members can get the package from the official Wettzell system monitoring page as tar-ball of the latest tagged version:
  - **tbd**
  - Change into the installation folder */home/oper/Software/* , download the package and unpack it with

```
tar -xzvf <package_version>.tar.gz
```

- Change into the new folder */home/oper/Software/mk6stationcode/make*
- Build the sources

```
make build
```

- Install the sources inclusively the startup script

```
make install
```

4.3.3 Installation without the Wettzell Mk6 station code

- Download Zabbix sources as described in 0) SysMon Node VLBI
- Unpack sources e.g. into a folder /usr2/prog/Software/

```
tar -xzvf zabbix_3.2.4.orig.tar.gz
```

- Change into newly created directory, run the configuration utility and buidl the agent

```
▪      mkdir Software
▪      mkdir mk6stationcode
▪      mkdir zabbix
▪      cd /usr2/prog/Software/mk6stationcode/zabbix/zabbix-3.2.4
▪      ./configure --enable-agent
▪      make
▪
```

- Make a safety copy of the original configuration file

```
▪      cp /usr2/prog/Software/mk6stationcode/zabbix/zabbix-
3.2.4/config/zabbix_agentd.conf
/usr2/prog/Software/mk6stationcode/zabbix/zabbix-
3.2.4/config/zabbix_agentd.conf_orig
▪
```

- Edit the configuration file and change the falues to the following (Hostname should be the name for the individual system which is also used later in the Zabbix front end; the IP addresses of the server *<zabbix_server_ip>* must be those from the real Zabbix server PC)

```
▪     LogFile=/tmp/zabbix_agentd.log
▪     DebugLevel=3
▪     Server=<zabbix_server_ip>
▪
```

- Become root rights and do the following steps

```
▪      su
▪      cp /usr2/prog/Software/mk6stationcode/zabbix/zabbix-
3.2.4/conf/zabbix_agentd.conf /usr/local/etc/zabbix_agentd.conf
▪      cp /usr2/prog/Software/mk6stationcode/zabbix/zabbix-
3.2.4/src/zabbix_agent/zabbix_agentd /usr/sbin/zabbix_agentd
▪      groupadd zabbix
▪      useradd -g zabbix zabbix
▪
```

- Test the start of the Zabbix agent

```
▪     /usr2/prog/Software/mk6stationcode/zabbix/zabbix-
3.2.4/src/zabbix_agent/zabbix_agentd -c
/usr2/prog/Software/mk6stationcode/zabbix/zabbix-
3.2.4/conf/zabbix_agentd.conf
▪
```

- Create a startscript in /etc/init.d/ e.g. with the name zabbix_agentd or include it to another start script. To start the zabbix_agentd it must contain this:

```
▪      #!/bin/sh
▪      ### BEGIN INIT INFO
▪      # Provides: zabbix_agentd
▪      # Required-Start:
▪      # Required-Stop:
▪      # Default-Start: 2 3 4 5
▪      # Default-Stop: 0 1 6
▪      # Short-Description: Start zabbix_agent for PC monitoring
▪      # Description: Start zabbix_agent for PC monitoring
▪      ### END INIT INFO
▪
▪      case $1 in
```

```
    start)
        su daemon -c /usr/bin/zabbix_agentd
        ;;
    stop)
        kill `cat /tmp/zabbix_agentd.pid`
        ;;
    restart)
        $0 stop
        sleep 2
        $0 start
        ;;
    *)
        echo "usage: $0 [start|stop|restart]"
        ;;
    esac

```

- **Attention: All scripts and programs used and described below must be installed manually to the right folders**

4.3.4 Configure Zabbix agent

- Use the configuration file from the Wettzell package which contains this

```
# This is a configuration file for Zabbix agent daemon (Unix)
# To get more information about Zabbix, visit http://www.zabbix.com
# This is a simplified version just for VLBI

LogFile=/tmp/zabbix_agentd.log
DebugLevel=3
Server=<zabbix_server_ip>



# ==========================================================
# = UserParameter ==========================================
#Test using: zabbix_get -s <mk6_ip_address> -k mk6.disk10_size
#            where
#            <mk6_ip_address> is the actual Mark6 IP address
# ==========================================================

# = MODULE 1 ===============================================

UserParameter=mk6.disk1_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1
UserParameter=mk6.disk1_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1
UserParameter=mk6.disk1_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1
UserParameter=mk6.disk1_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1

UserParameter=mk6.disk10_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 0
```

- UserParameter=mk6.disk10_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 0
- UserParameter=mk6.disk10_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1 0
- UserParameter=mk6.disk10_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1 0
-
- UserParameter=mk6.disk11_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 1
- UserParameter=mk6.disk11_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 1
- UserParameter=mk6.disk11_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1 1
- UserParameter=mk6.disk11_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1 1
-
- UserParameter=mk6.disk12_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 2
- UserParameter=mk6.disk12_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 2
- UserParameter=mk6.disk12_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1 2
- UserParameter=mk6.disk12_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1 2
-
- UserParameter=mk6.disk13_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 3
- UserParameter=mk6.disk13_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 3
- UserParameter=mk6.disk13_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1 3
- UserParameter=mk6.disk13_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1 3
-
- UserParameter=mk6.disk14_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 4
- UserParameter=mk6.disk14_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 4
- UserParameter=mk6.disk14_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1 4
- UserParameter=mk6.disk14_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1 4
-
- UserParameter=mk6.disk15_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 5
- UserParameter=mk6.disk15_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 5
- UserParameter=mk6.disk15_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 1 5
- UserParameter=mk6.disk15_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 1 5
-
- UserParameter=mk6.disk16_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 1 6
- UserParameter=mk6.disk16_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 1 6

```
▪ UserParameter=mk6.disk16_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 1 6
▪ UserParameter=mk6.disk16_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 1 6
▪
▪ UserParameter=mk6.disk17_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 1 7
▪ UserParameter=mk6.disk17_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 1 7
▪ UserParameter=mk6.disk17_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 1 7
▪ UserParameter=mk6.disk17_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 1 7
▪
▪ # = MODULE 2 ================================================
▪
▪ UserParameter=mk6.disk2_size,/home/oper/Software/mk6stationcode/bin/mk6_che
  ckvolume.sh Size 2
▪ UserParameter=mk6.disk2_used,/home/oper/Software/mk6stationcode/bin/mk6_che
  ckvolume.sh Used 2
▪ UserParameter=mk6.disk2_avail,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Avail 2
▪ UserParameter=mk6.disk2_used_percent,/home/oper/Software/mk6stationcode/bin
  /mk6_checkvolume.sh Use-percent 2
▪
▪ UserParameter=mk6.disk20_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 2 0
▪ UserParameter=mk6.disk20_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 2 0
▪ UserParameter=mk6.disk20_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 2 0
▪ UserParameter=mk6.disk20_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 2 0
▪
▪ UserParameter=mk6.disk21_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 2 1
▪ UserParameter=mk6.disk21_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 2 1
▪ UserParameter=mk6.disk21_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 2 1
▪ UserParameter=mk6.disk21_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 2 1
▪
▪ UserParameter=mk6.disk22_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 2 2
▪ UserParameter=mk6.disk22_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 2 2
▪ UserParameter=mk6.disk22_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 2 2
▪ UserParameter=mk6.disk22_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 2 2
▪
▪ UserParameter=mk6.disk23_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 2 3
▪ UserParameter=mk6.disk23_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 2 3
```

- UserParameter=mk6.disk23_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 2 3
- UserParameter=mk6.disk23_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 2 3
-
- UserParameter=mk6.disk24_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 2 4
- UserParameter=mk6.disk24_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 2 4
- UserParameter=mk6.disk24_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 2 4
- UserParameter=mk6.disk24_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 2 4
-
- UserParameter=mk6.disk25_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 2 5
- UserParameter=mk6.disk25_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 2 5
- UserParameter=mk6.disk25_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 2 5
- UserParameter=mk6.disk25_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 2 5
-
- UserParameter=mk6.disk26_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 2 6
- UserParameter=mk6.disk26_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 2 6
- UserParameter=mk6.disk26_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 2 6
- UserParameter=mk6.disk26_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 2 6
-
- UserParameter=mk6.disk27_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 2 7
- UserParameter=mk6.disk27_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 2 7
- UserParameter=mk6.disk27_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 2 7
- UserParameter=mk6.disk27_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 2 7
-
- # = MODULE 3 ==========
-
- UserParameter=mk6.disk3_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3
- UserParameter=mk6.disk3_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3
- UserParameter=mk6.disk3_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3
- UserParameter=mk6.disk3_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3
-
- UserParameter=mk6.disk30_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 0
- UserParameter=mk6.disk30_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 0

- UserParameter=mk6.disk30_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 0
- UserParameter=mk6.disk30_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3 0
-
- UserParameter=mk6.disk31_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 1
- UserParameter=mk6.disk31_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 1
- UserParameter=mk6.disk31_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 1
- UserParameter=mk6.disk31_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3 1
-
- UserParameter=mk6.disk32_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 2
- UserParameter=mk6.disk32_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 2
- UserParameter=mk6.disk32_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 2
- UserParameter=mk6.disk32_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3 2
-
- UserParameter=mk6.disk33_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 3
- UserParameter=mk6.disk33_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 3
- UserParameter=mk6.disk33_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 3
- UserParameter=mk6.disk33_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3 3
-
- UserParameter=mk6.disk34_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 4
- UserParameter=mk6.disk34_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 4
- UserParameter=mk6.disk34_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 4
- UserParameter=mk6.disk34_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3 4
-
- UserParameter=mk6.disk35_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 5
- UserParameter=mk6.disk35_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 5
- UserParameter=mk6.disk35_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 5
- UserParameter=mk6.disk35_used_percent,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Use-percent 3 5
-
- UserParameter=mk6.disk36_size,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Size 3 6
- UserParameter=mk6.disk36_used,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Used 3 6
- UserParameter=mk6.disk36_avail,/home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh Avail 3 6

```
▪ UserParameter=mk6.disk36_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 3 6
▪
▪ UserParameter=mk6.disk37_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 3 7
▪ UserParameter=mk6.disk37_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 3 7
▪ UserParameter=mk6.disk37_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 3 7
▪ UserParameter=mk6.disk37_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 3 7
▪
▪ # = MODULE 4 ==============================================
▪
▪ UserParameter=mk6.disk4_size,/home/oper/Software/mk6stationcode/bin/mk6_che
  ckvolume.sh Size 4
▪ UserParameter=mk6.disk4_used,/home/oper/Software/mk6stationcode/bin/mk6_che
  ckvolume.sh Used 4
▪ UserParameter=mk6.disk4_avail,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Avail 4
▪ UserParameter=mk6.disk4_used_percent,/home/oper/Software/mk6stationcode/bin
  /mk6_checkvolume.sh Use-percent 4
▪
▪ UserParameter=mk6.disk40_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 0
▪ UserParameter=mk6.disk40_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 0
▪ UserParameter=mk6.disk40_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 0
▪ UserParameter=mk6.disk40_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 0
▪
▪ UserParameter=mk6.disk41_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 1
▪ UserParameter=mk6.disk41_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 1
▪ UserParameter=mk6.disk41_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 1
▪ UserParameter=mk6.disk41_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 1
▪
▪ UserParameter=mk6.disk42_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 2
▪ UserParameter=mk6.disk42_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 2
▪ UserParameter=mk6.disk42_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 2
▪ UserParameter=mk6.disk42_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 2
▪
▪ UserParameter=mk6.disk43_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 3
▪ UserParameter=mk6.disk43_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 3
▪ UserParameter=mk6.disk43_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 3
```

```
- UserParameter=mk6.disk43_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 3
-
- UserParameter=mk6.disk44_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 4
- UserParameter=mk6.disk44_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 4
- UserParameter=mk6.disk44_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 4
- UserParameter=mk6.disk44_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 4
-
- UserParameter=mk6.disk45_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 5
- UserParameter=mk6.disk45_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 5
- UserParameter=mk6.disk45_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 5
- UserParameter=mk6.disk45_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 5
-
- UserParameter=mk6.disk46_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 6
- UserParameter=mk6.disk46_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 6
- UserParameter=mk6.disk46_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 6
- UserParameter=mk6.disk46_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 6
-
- UserParameter=mk6.disk47_size,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Size 4 7
- UserParameter=mk6.disk47_used,/home/oper/Software/mk6stationcode/bin/mk6_ch
  eckvolume.sh Used 4 7
- UserParameter=mk6.disk47_avail,/home/oper/Software/mk6stationcode/bin/mk6_c
  heckvolume.sh Avail 4 7
- UserParameter=mk6.disk47_used_percent,/home/oper/Software/mk6stationcode/bi
  n/mk6_checkvolume.sh Use-percent 4 7
```

- Adapt the server address "*Server=<zabbix_server_ip>*" to the actually used one of Zabbix proxy or Zabbix server
- The configuration uses the script */home/oper/Software/mk6stationcode/bin/mk6_checkvolume.sh* to read the values from the Mark6:

```
- #!/bin/bash
- #*********************************************************************/
- #*! \file
- #   \brief Script to return the values of the volume info of single disks
  <br>
- #
- #*******************************************************************
- # Dependencies: -
- #*********************************************************************/
-
- #*******************************************************************
- # SVN: Version Control with Subversion
```

```
# ------------------------------------------------------------------
# $Id$
#******************************************************************
# LICENSE AND WARRANTY INFORMATION (FOR THE GENERATED SOURCE CODE)
# ================================================================
# Copyright (C) 2017
# Forschungseinrichtung Satellitengeodaesie, TU Muenchen
# and Bundesamt fuer Kartographie und Geodaesie
# Geodetic Observatory Wettzell
# Sackenrieder Str. 25
# D-93444 Bad Koetzting
# Germany
# (and the developers,
#  mainly A. Neidhardt, Ch. Ploetz)
#
# This program is FREE SOFTWARE under the terms of GNU Lesser General
# Public License v3 (or any later version) and may be used following
# this definitions as published by the Free Software Foundation at
# http://www.gnu.de/documents/lgpl-3.0.en.html. Software parts which
# include elements from external software distributions may be under
# different licenses as the Sun License/BSD License for the ONC/Sun RPC
# (http://www.opensource.org/licenses/bsd-license.php)
# and the wxWindows Library Licence for the GUI parts with wxWidgets
# (http://www.opensource.org/licenses/wxwindows.php).
# In case of variations to the above licenses each particular developer
# is responsible for defining the dedicated license conditions and terms.
#
# This program is distributed in the hope that it will be useful.
# IT IS PROVIDED AS IT IS WITH NO WARRANTIES OF ANY KIND INCLUDING
# THE WARRANTIES OF DESIGN, MERCHANTIBILITY AND FITNESS FOR A
# PARTICULAR PURPOSE, OR ARISING FROM A COURSE OF DEALING, USAGE
# OR TRADE PRACTICE.
#
# The software is provided with no support and without any obligation
# on the part of the Geodetic Observatory Wettzell to assist in its
# use, correction, modification or enhancement. THE Geodetic Observatory
# Wettzell SHALL HAVE NO LIABILITY WITH RESPECT TO THE INFRINGEMENT OF
# COPYRIGHTS, TRADE SECRETS OR ANY PATENTS BY THE SOFTWARE OR ANY PART
# THEREOF. In no event will the Geodetic Observatory Wettzell be liable
# for any lost revenue or profits or other special, indirect and
# consequential damages, even if the Geodetic Observatory Wettzell
# has been advised of the possibility of such damages.
#
# You should have received a copy of the license(s) along with this
program.
#******************************************************************/

if [ $# -lt 2 ] ||  [ $# -gt 3 ] ; then
    echo "./mk6_checkvolume.sh <value_type> <module_num> [<disk_num>]"
    echo "     Returns the vilume data definded in <value_type>"
    echo "     <value_type>      : which value should be returned:
Filesystem | Size |  Used | Avail | Use-percent"
    echo "     <module_num>      : MK6 module number 1, 2, 3, or 4"
    echo "     <disk_num>        : disk number on MK6 module 0, 1, 2, 3, 4,
5, 6, or 7 (optional)"
    exit 1
```

```
fi

MODULE=$2
DISK=$3

VALUE=""

if [ $# -eq 2 ] ; then
    DISK=0
    SUMVALUE=0
    ARGUMENT_OK=1
    if [ "$1" = "Filesystem" ] ; then
        VALUE=""
        exit;
    fi
    while [  $DISK -le 7 ]; do
        DISKINFO=`/bin/df -BG /mnt/disks/${MODULE}/${DISK} | /bin/grep "/mnt/disks/${MODULE}/${DISK}"`;
        if [[ ! -z $DISKINFO ]]; then
            if [ "$1" = "Size" ] ; then
                VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9]\+\)%.*/\2/g'`
                VALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}/1024}"`
            elif [ "$1" = "Used" ] ; then
                VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9]\+\)%.*/\3/g'`
                VALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}/1024}"`
            elif [ "$1" = "Avail" ] ; then
                VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9]\+\)%.*/\4/g'`
                VALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}/1024}"`
            elif [ "$1" = "Use-percent" ] ; then
                VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9]\+\)%.*/\5/g'`
            else
                VALUE=""
                ARGUMENT_OK=0
            fi
        fi
        let DISK=DISK+1
        SUMVALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}+${SUMVALUE}}"`
    done
    if [ $ARGUMENT_OK -eq 0 ] ; then
        VALUE=""
    else
        VALUE=${SUMVALUE};
    fi
else
    DISKINFO=`/bin/df -BG /mnt/disks/${MODULE}/${DISK} | /bin/grep "/mnt/disks/${MODULE}/${DISK}"`;
    if [ -z "$DISKINFO" ] ; then
    VALUE=""
```

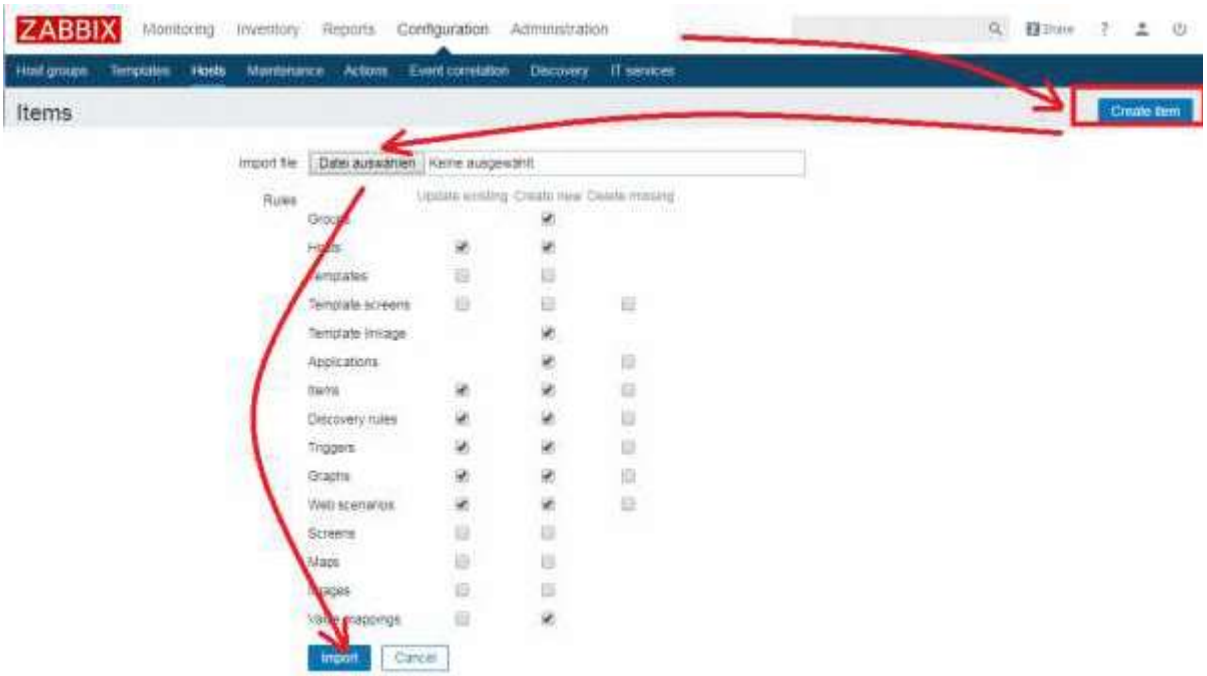
```
▪      else
▪          if [ "$1" = "Filesystem" ] ; then
▪              VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[ ]\+\([0-
  9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+
▪ \([0-9]\+\)%.*/\1/g'`
▪          elif [ "$1" = "Size" ] ; then
▪              VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[ ]\+\([0-
  9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+
▪ \([0-9]\+\)%.*/\2/g'`
▪              VALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}/1024}"`
▪          elif [ "$1" = "Used" ] ; then
▪              VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[ ]\+\([0-
  9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+
▪ \([0-9]\+\)%.*/\3/g'`
▪              VALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}/1024}"`
▪          elif [ "$1" = "Avail" ] ; then
▪              VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[ ]\+\([0-
  9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+
▪ \([0-9]\+\)%.*/\4/g'`
▪              VALUE=`awk "BEGIN {printf \"%.2f\n\", ${VALUE}/1024}"`
▪          elif [ "$1" = "Use-percent" ] ; then
▪              VALUE=`/bin/echo ${DISKINFO} | /bin/sed -e 's/\(.*\)[ ]\+\([0-
  9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+\([0-9\.]\+\)[TMG][ ]\+
▪ \([0-9]\+\)%.*/\5/g'`
▪          else
▪              VALUE=""
▪          fi
▪      fi
▪  fi
▪  echo ${VALUE}
▪
▪
▪  #*********************************************************************
▪  # END OF FILE (SVN: Version Control with Subversion)
▪  # ------------------------------------------------------------------
▪  # $Id$
▪  #*********************************************************************/
```

4.3.5 Configure Zabbix server

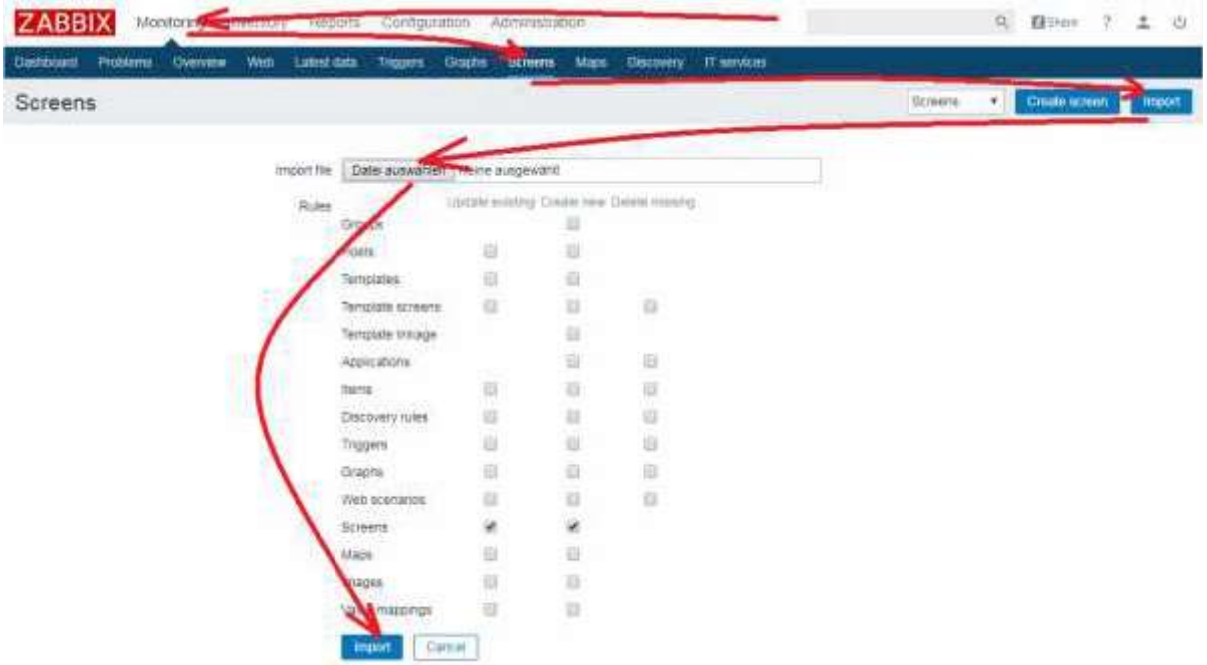
- It is necessary to configure a host, graphs and screens on the ZABBIX server for the monitoring of the data in the monitoring data center

4.3.6 Simple configuration using the Wettzell template files

- The simple way is to use the template files created at Wettzell. To do this, follow the following description. The templates can be requested from the Wettzell observatory.
- Before you import the template file, you can change the IP addresses and naming in the XML file using a text editor. The current Mark6 host uses a passive ZABBIX proxy to pass firewalls.
- Import the host template: 20180406_zbx_host_mk61ttw2_via_proxy.xml



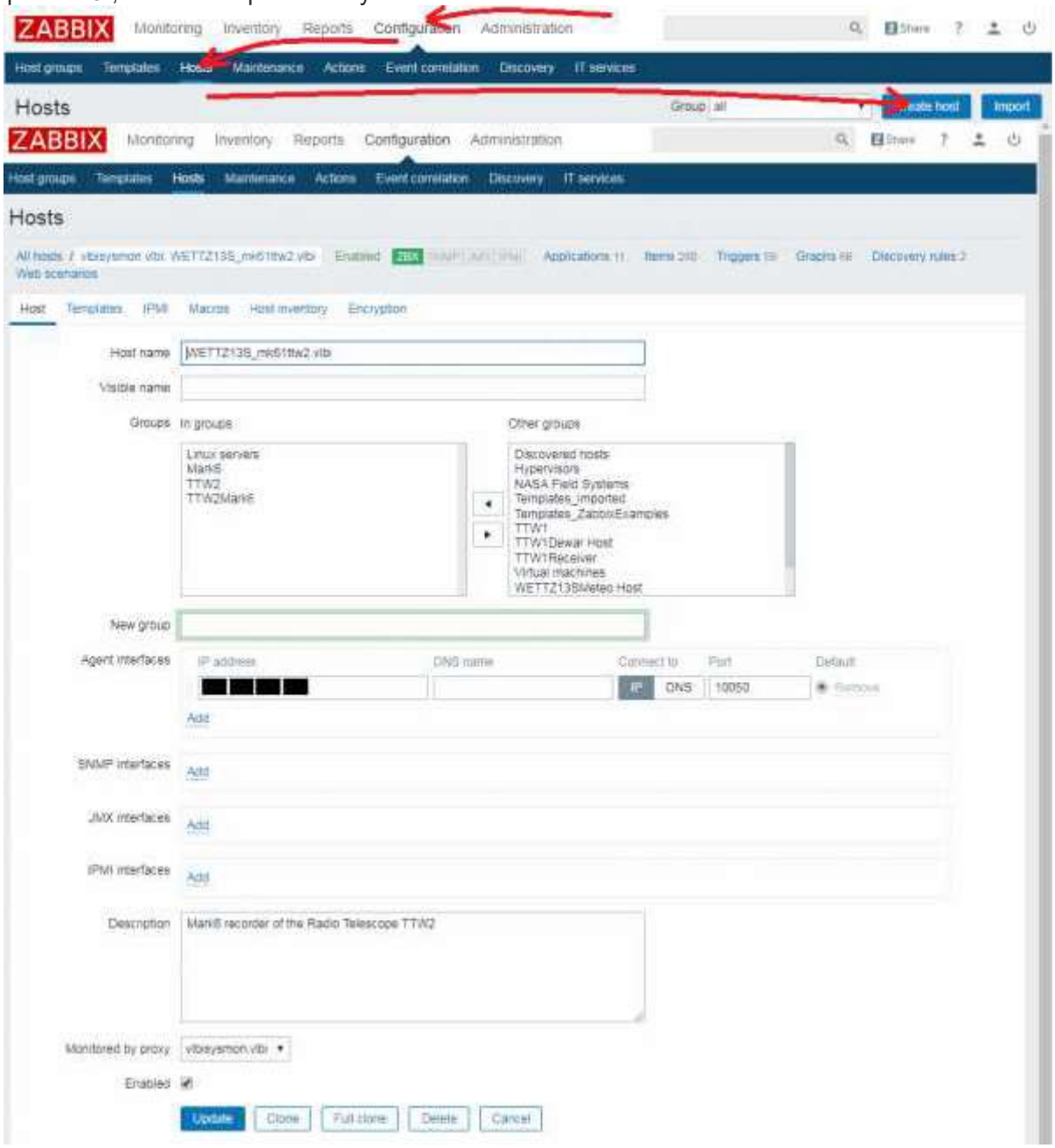
-
- Import the screen template: [20180406_zbx_screen_mk6.xml]



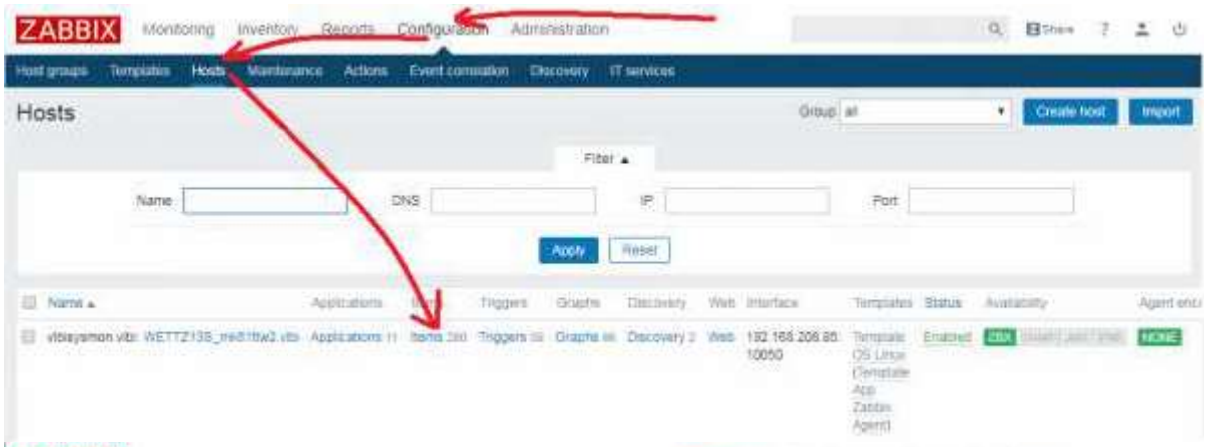
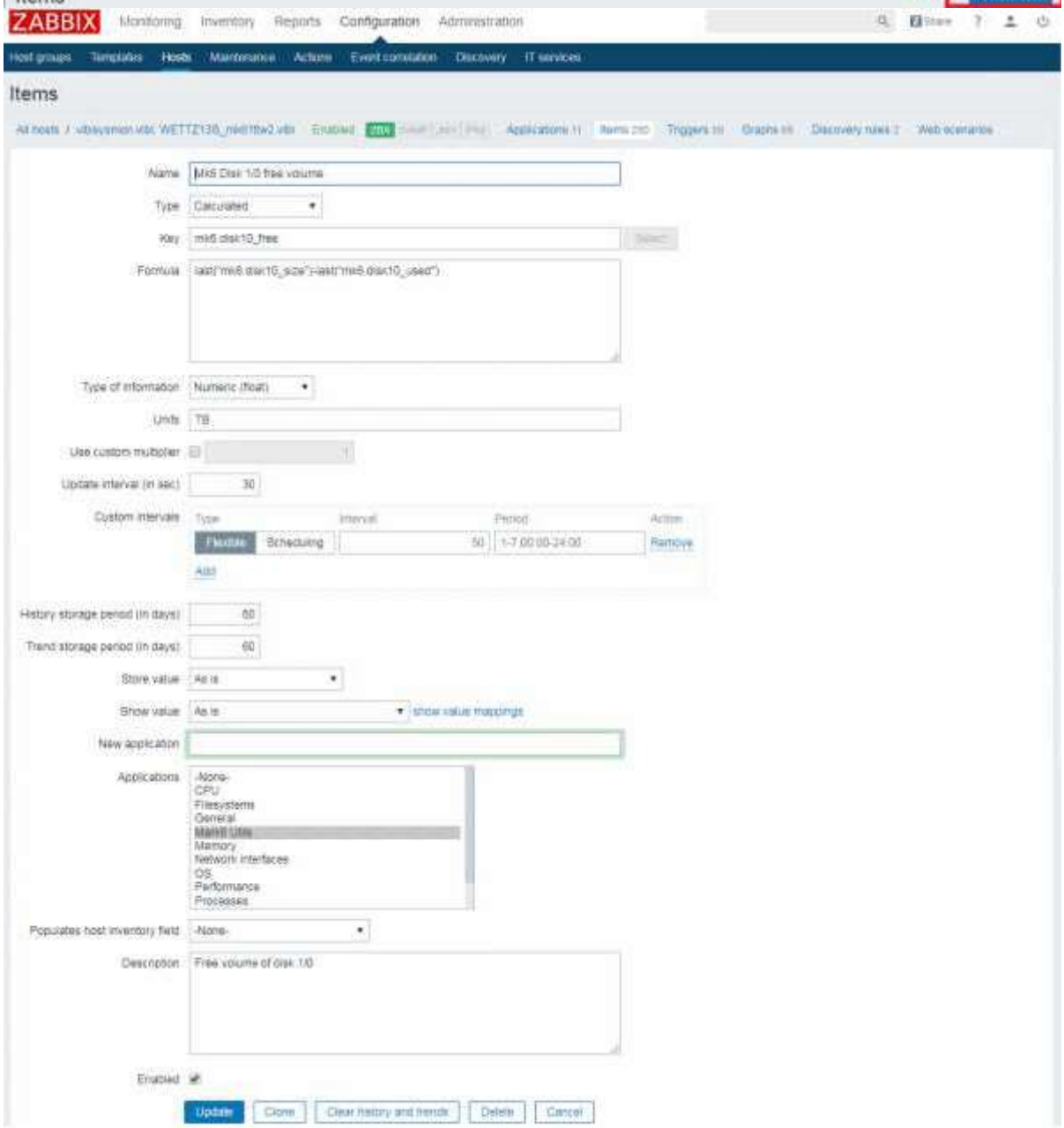
-
- You should now be able to open the following screen:


ZABBIX
Screens
Trigger info
Trigger overview

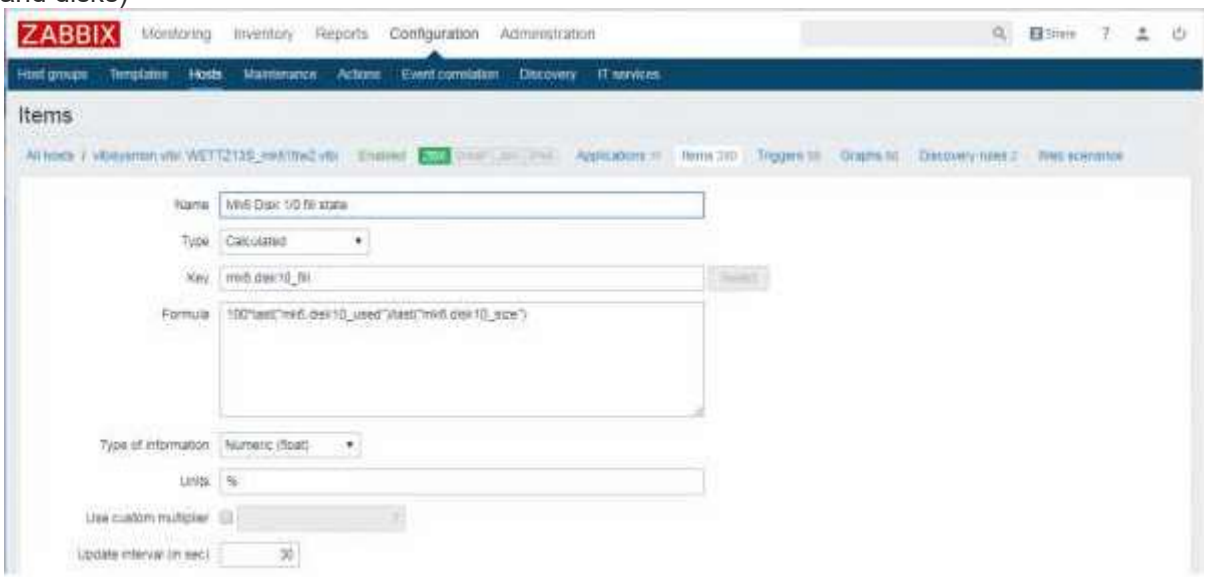
4.3.7 Manual configuration without the Wettzell template files

- **Create a host** representation for the Mark6. Assign the group “Linux servers” and create a new group “Mark6”, and other specific for your institute
- 

- **Create new items** according to the items defined above in the agent configuration (useful additional items are free space on Mark6 modules and disks, used space on modules and disks, total space on modules and disks, etc.)

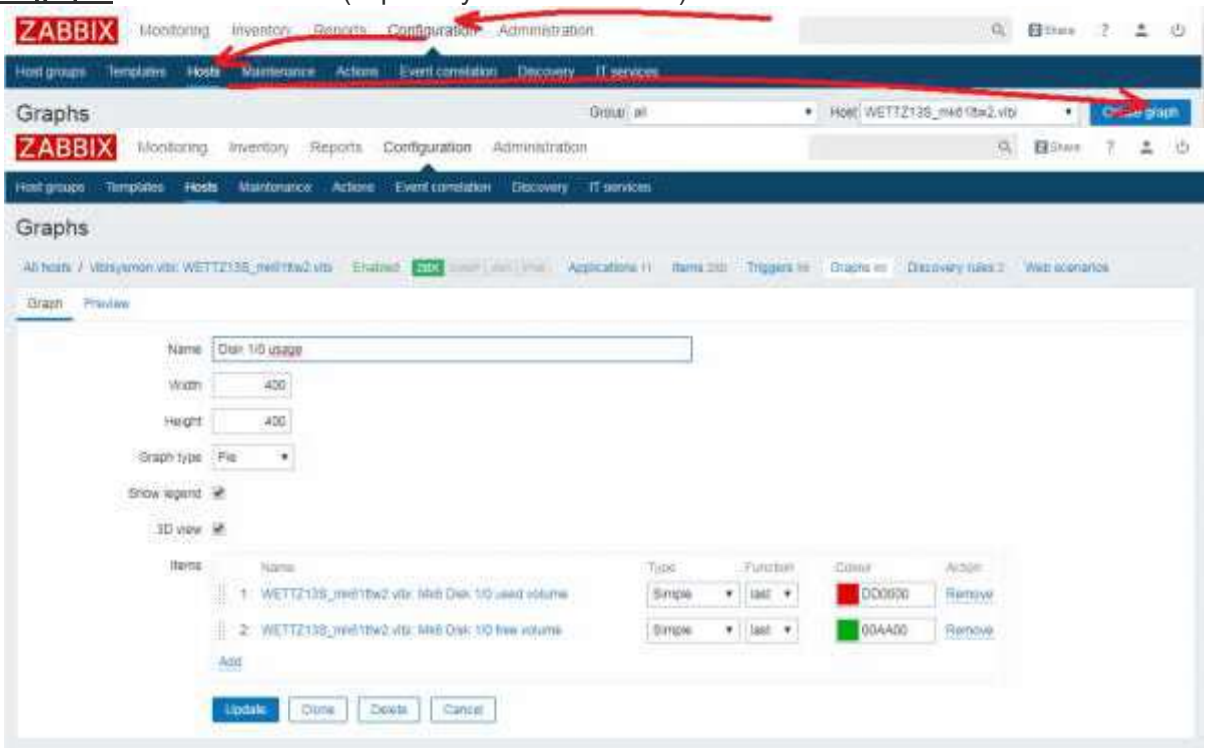
- 
ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
Host groups
Templates
Hosts
Maintenance
Actions
Event correlation
Discovery
IT services
Hosts
Group all
Create host
Import
Filter
Name
DNS
IP
Port
Apply
Reset
Name
Applications
Items
Triggers
Graphs
Discovery
Web
Interface
Templates
Status
Availability
Agent enc
Enabled
NONE
- 
ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
Host groups
Templates
Hosts
Maintenance
Actions
Event correlation
Discovery
IT services
Items
Create item
- 
ZABBIX
Monitoring
Inventory
Reports
Configuration
Administration
Host groups
Templates
Hosts
Maintenance
Actions
Event correlation
Discovery
IT services
Items
Name
Type
Calculated
Key
Formula
Type of information
Numeric (float)
Units
TB
Use custom multiplier
Update interval (in sec)
30
Custom intervals
Type
Interval
Period
Action
Flexible
Scheduling
50
1-7,00:00-24:00
Remove
Add
History storage period (in days)
60
Trend storage period (in days)
60
Store value
As is
Show value
As is
show value mappings
New application
Applications
-None-
CPU
Filesystems
General
Memory
Network interfaces
OS
Performance
Processes
Populates host inventory field
-None-
Description
Free volume of disk 1/0
Enabled
Update
Clone
Clear history and trends
Delete
Cancel

- It is also possible to combine items to create a new one (like the fill state of the modules and disks)

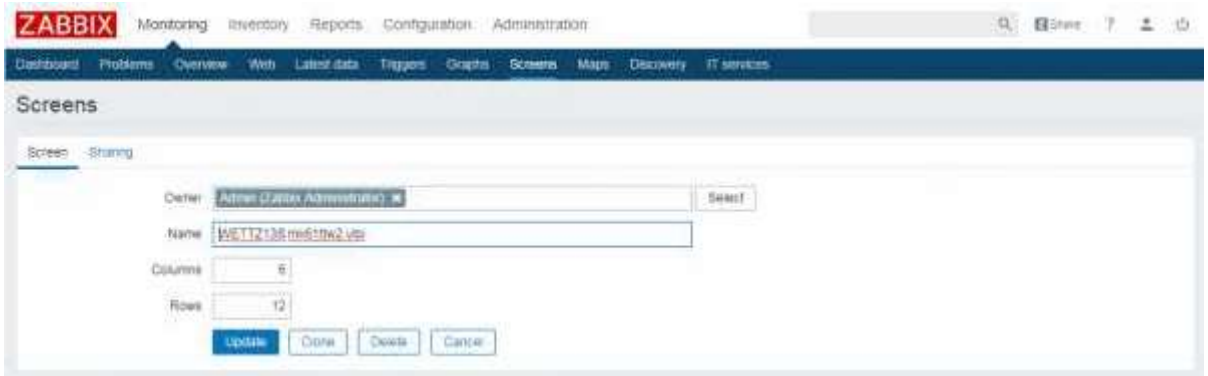


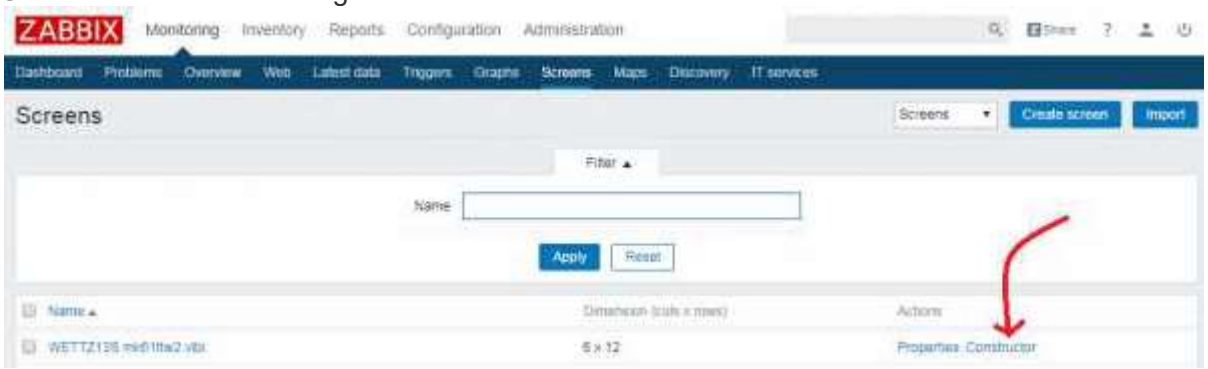
- ...

- **Create graphs** for the new items (especially for the fill states)



- **Create screen** for the new host

- 

- Use the constructor to design the screen
- 

- How to add elements is described here: https://www.zabbix.com/documentation/3.0/manual/config/visualisation/screens

4.4 Appendix: Installation and configuration of the monitoring of an SNMP device (like a USP)

Written by M. Schönberger

4.4.1 Prepare the server and agent for SNMP

4.4.1.1 SNMP-traps

- Zabbix is not able to receive SNMP-traps directly. It needs help from other tools:

```
sudo apt-get install snmp snmpd snmptt snmptrapd snmp-mibs-downloader
```

- Default is to run the agent snmpd, we need the snmptrapd running. Change configuration file ***/etc/default/snmpd***

```
SNMPDRUN=no
```

In file ***/etc/default/snmptrapd*** change following line:

```
TRAPDRUN=yes
```

- To get readable traps, trapper deamon must pass the received traps to trap translate deamon. Therefore edit configuration file ***/etc/snmp/snmptrapd.conf***

```
traphandle default /usr/sbin/snmptt
disableAuthorization yes
```

- To get zabbix conform messages edit following lines in file ***/etc/snmp/snmptt.ini***

```
mode = standalone
translate_log_trap_oid = 2
net_snmp_perl_enable = 1
date_time_format = %H:%M:%S %Y/%m/%d
log_file = /tmp/zabbix_traps.tmp
log_system_enable = 1
mibs_environment = ALL
```

- Make backup of original file /etc/snmp/snmptt.conf and change the file to only containing following two lines:

```
EVENT general .* "General event" Normal
FORMAT ZBXTRAP $aA $ar severity:$s $Fn$+*
```

- After editing the config files we have to restart the services

Till Ubuntu 14.04:

```
service snmpd restart && service snmptt restart
```

Since Ubuntu 16.04:

```
systemctl restart snmpd.service && systemctl restart snmptrapd.service &&
systemctl restart snmptt.service
```

- Configurate and restart zabbix-server

```
sudo nano /usr/local/etc/zabbix_server.conf
```

Delete # before following lines

```
SNMPTrapperFile=/tmp/zabbix_traps.tmp
```

```
StartSNMPTrapper=1
```

Restart server:

```
systemctl restart zabbix-server.service
```

- Configure logrotate for /tmp/zabbix_traps.tmp

To prevent the trapper-file from growing to big we can use the linux tool logrotate. It should be already installed. Create configuration file

```
sudo nano /etc/logrotate.d/zabbix_traps
```

and insert following lines

```
/tmp/zabbix_traps.tmp {
   daily
   size 10M
   compress
   notifempty
   missingok
   maxage 365
   rotate 1
}
```

4.4.1.2 Configure SNMP monitoring of UPS with zabbix

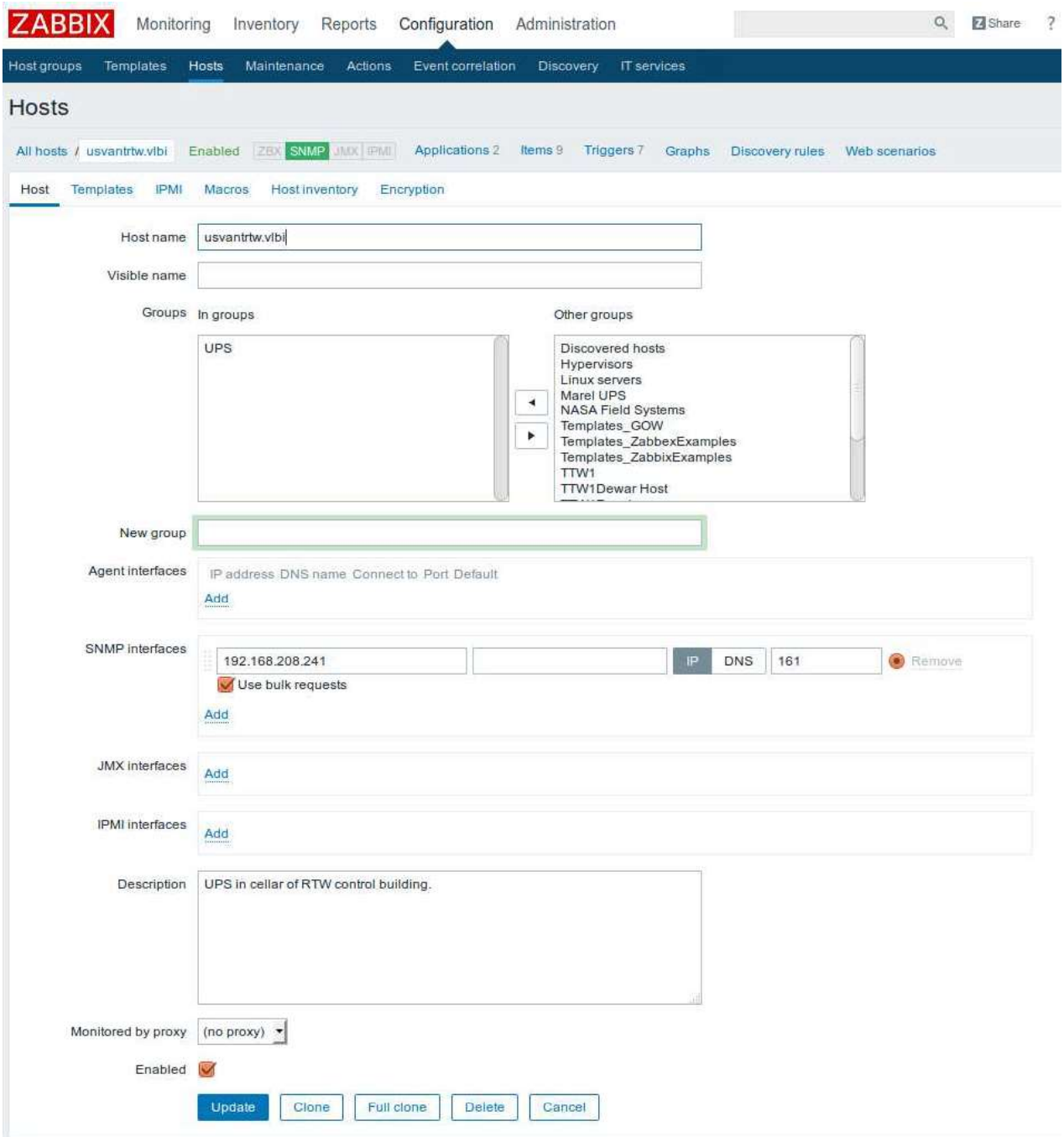
- Configurate SNMP for Twin- and RTW-UPS

Open web interface of UPS (http://192.168.208.240) in a web browser and log in. Go to Cofiguration→SNMP and set at least 192.168.208.235 (sysmonvlbi.vlbi - the vlbi zabbix server) as SNMP Trap Receiver. Set community as well.

Click Apply and Configuration→Save Configuration. Save and reboot. Do the same for usvantrtw.vlbi (http://192.168.208.241)

- Add hosts and Link with the snmp ups templates

Login to web interface of Zabbix on http://192.168.208.235 Go to Configuration→Hosts. At right upper corner click Create host.



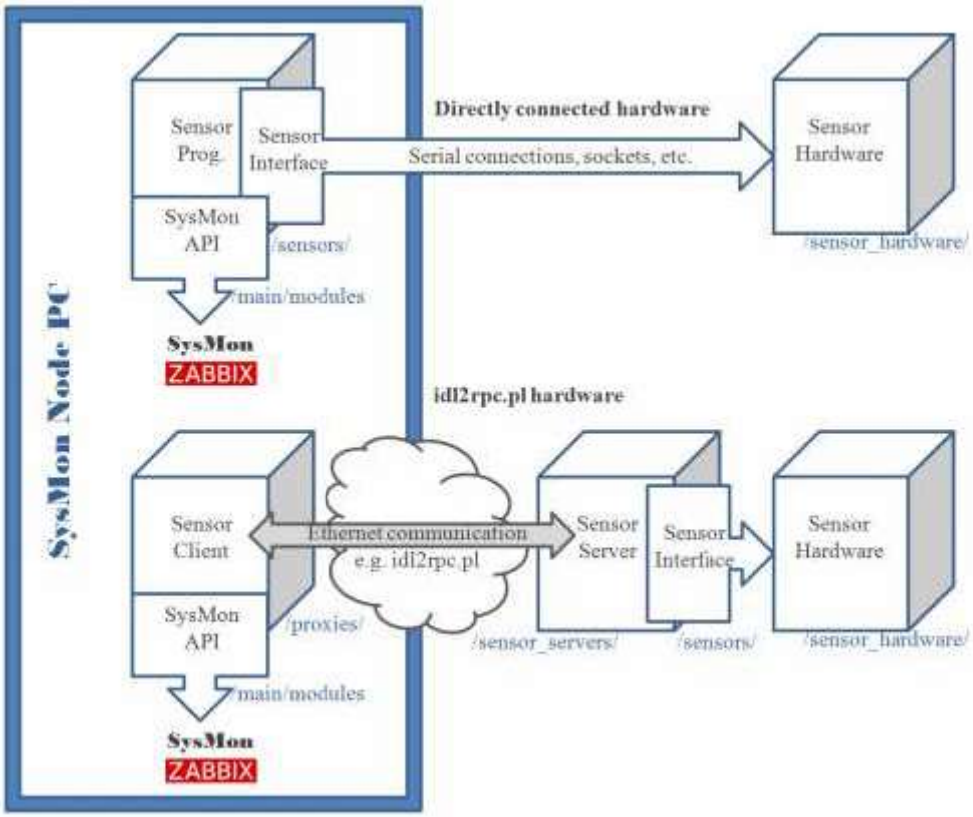
Insert host name and group UPS. Add a SNMP interface. The agent interface can be removed. Insert description and click Update.

4.5 Appendix: Installation and configuration of the monitoring with a SysMon node (with updates to D8.4)

The data from other equipment at location of the telescope are interesting for analysis and not only for operations. Therefore, tey are **analysis data**which are saved to be used by the VLBI analysis centers. Analysis data are managed by SysMon and in the Zabbix system. There is a web archive containing the data history over years in individual formats on the system monitoring PC.
The management of the analysis data requires a suitable installation of the Wettzell System Monitoring Suite (SysMon): see 0) SysMon Node VLBI.

- The SysMon suite is maintained by the Wettzell observatory (see http://xsamba.wtz/svn/vlbi/trunk/code/vlbisysmon/) and consists of the following parts, while for own programs only the folder "main" is essential.

-
  - **main:** contains all central modules and programs of the current SysMon suite, e.g.
    - **modules:** with all the C/C++ modules of the API
    - **sysmon_backup:** a program which can be used to backup content from the SysMon database into files
    - **sysmon_sender:** a program which can be used in scripts and other programs to register sensors and to send data to SysMon using system calls
  - **proxies:** Proxies are predefined clients, which fetch data from already existing sensor servers (mainly for Wettzell observatory needs in the form of idl2rpc.pl clients)
  - **sensor_hardware:** Programs which run on sensor hardware like microcontrollers
  - **sensor_servers:** Server programs (mainly for Wettzell observatory needs in the form of idl2rpc.pl servers) which fetch data from sensors, process them and offer them to clients.
  - **sensor:** Modules and test programs which realize the interface to a sensor hardware
- The SysMon node PCs use the following software components (PPT-Images collection):



-
- The SysMon sender or SysMon API must always be used on the PC where SysMon and Zabbix is running. A remote data injection is not enabled. Therefore, proxies can be used.

4.5.1 Create an own C/C++ program to send in data of a dedicated sensor control point or use a script calling "sysmon_senderc"

- All Wettzell programs are located in the SysMon project. Other external programs just need the files from directory “main” of the SysMon suite.
- A simple program just consists of the following scheme:

```
....
#include "mcidb_access.hpp"
....

int main (int argc, char * argv[])
{
    /// <b> Variables <b>
    unsigned short usError = 0;
    mcidb_access CSysMonAPI;  /// CSysMonAPI = object to access the Wettzell System Monitoring SysMon

    /// Check program arguments where a configuration file should be handed over somehow
    ...

    /// Register new sensor control point using a predefined configuration file
    if (CSysMonAPI.usRegisterSensors(argv[1]))
    {
```

```
        std::cout << "[ERROR] Cannot register sensors\n";
        usError = 2;
        goto FinishProgram;
    }

    while (true)
    {
        double dPressureMBar = 0.0;    /// dPressureMBar = pressure value of a dewar
        std::stringstream ssValue;     /// ssValue = conversion of double values to strings
        ...

        /// Open connection to sensor hardware
        ...

        /// Read data from sensor, e.g a pressure value of the dewar "dPressureMBar"
        ...

        /// Close connection to sensor hardware
        ...

        /// Send data to the system monitoring SysMon, e.g. a pressure value of the dewar "dPressureMBar"
        ssValue << std::fixed << std::setprecision(7) <<  dPressureMBar * 10000;  // =0.0001000*10^-4 mbar
        if (CSysMonAPI.usSendSingleData ("TTW1Dewar_Pressure", ssValue.str()))
        {
            std::cerr << "[ERROR] Cannot send TTW1Dewar_Pressure to SysMon\n";
        }
        ssValue.str("");
        ...

        /// Manage own timing interval
        sleep (60);
    }

FinishProgram:
    /// Error processing
    ...

    return 0;
}

```

- Scripts can also just call the program “sysmon_senderc” which is located at “*/home/oper/Software/vlbisysmon/main/sysmon_sender/bin/* ” using one of the following arguments:

```
    sysmon_send <-option>
    where <-option> is:
        -I configfile.conf (register)
        -R configfile.conf (deregister)
        -D configfile.conf (delete)
```

```
-d configfile.conf netsensorid (delete netsensorid)
-S configfile.conf (register, update, write, send)
-s configfile.conf netsensorid value [alarmlevel] (write, send)
-f configfile.conf datafile.txt (write, send)
```

4.5.2 Register the sensor control point at SysMon

- It is necessary to create a standardized SysMon configuration file which describes the new sensor control point with its sensors to register it in the SysMon system.
- A basic explanation of the configuration can be found in the Monitoring and Control Infrastructure Whitepaper: 20120323mciworkingdocument.pdf
- An example of such a configuration looks like the following file “dewarproxyc.conf” describing the dewar values of the Wn antenna at Wettzell:

```
<MCISensorControlPoint>
        ControlPointID                  = TTW1Dewar
        ControlPointType                = Proxie
        ControlPointPort                = 52700
    <MCISensorProprietarySettings>
        <Communication>
#            <IDL2RPCServer> # not yet implemented
#                SocketConnect           = 1                        #Flag for Connection
#                IPAddress               = 192.168.208.13          #IP Address
#                Port                    = 52666                   #PortNumber
#            </IDL2RPCServer>
            <IPServer>
                SocketConnect           = 1                        #Flag for Connection
                IPAddress               = 192.168.208.13           #IP Address
                Port                    = 52666                    #PortNumber
            </IPServer>
            <SerialConnection>
                SerialConnect           = 0                        #Flag for Connection
                tty                     = /dev/ttyS0               #Serial Port tty
                FDtty                   = 3                        #File Descriptor for tty
            </SerialConnect>
            <Settings>
                Timeout                 = 3                        #For both connections (sec)
            </Settings>
        </Communication>
        <WriteSensorData>
            FilePath                    = /var/www/html/monitoring_archive #Path of FileArchive
```

```
▪             WriteTime                   = 5
#Write every 1,2,3,4,5,6,10,12,15,20,30 or 60 minutes
▪         </WriteSensorData>
▪         <WriteZabbixData>
▪             Execution                   = yes                       #yes/no
Create file and send to zabbix
▪             FilePath                    = /tmp                      #Path of
zabbix_file.txt
▪             FileName                    = zabbix_file.txt
▪             ServiceFilePath             = /usr/bin/zabbix_sender   #for the
shell-command
▪         </WriteZabbixData>
▪         <CreateZabbixImportTemplates>
▪             Execution                   = yes
▪             FilePath                    =
/home/oper/Software/vlbisysmon/proxies/rxmon/xml              #Path of
zabbix_import_template and _host.txt
▪             FileNameHostTemplate        = zabbix_import_host.xml
▪         </CreateZabbixImportTemplates>
▪     </MCISensorProprietarySettings>
▪     <MCIZabbixConnection>
▪         Host                            = 127.0.0.1
▪         Port                            = 5432
▪         DBName                          = zabbix
▪         UserName                        = zabbix
▪         Pwd                             = zabbix
▪         Timeout                         = 3
▪     </MCIZabbixConnection>
▪     <MCIDBConnection>
▪         Host                            = 127.0.0.1
▪         Port                            = 5432
▪         DBName                          = sysmon
▪         UserName                        = sysmon
▪         Pwd                             = +sysmon!
▪         Timeout                         = 3
▪     </MCIDBConnection>
▪     <MCIBackupSettings>
▪         Execution                       = yes                       #yes/no
Create backupfile and delete from database tables
▪         ArchiveDays                     = 30                        #Interval
of backups (days)
▪         BackupPath                      = /tmp                      #Path for
TableBackups
▪         ServiceBackupPath               =
/home/oper/Software/vlbisysmon/main/sysmon_backup/bin/sysmon_backupc
#Path for the module sysmon_backup
▪     </MCIBackupSettings>
▪     <MCISensor>
▪             SensorID                    = TTW1Dewar_TempAmbient
▪             SensorName                  = TempAmbient
▪             SensorUnit                  = deg C
▪             SensorManufacturer          = X
▪             SensorModel                 = X
▪             SensorPosition              = X
▪             SensorUpdateInterval        = X
▪             SensorResolution            = X
```

```
▪               SensorDataAvailabilityTime = X
▪               SensorMinLimit             = X
▪               SensorMaxLimit             = X
▪               SensorMinWarningLimit      = 5
▪               SensorMaxWarningLimit      = 35
▪               SensorMinAlertLimit        = 0
▪               SensorMaxAlertLimit        = 40
▪               SensorFlagProvider         = yes
▪               SensorFlagConsumer         = no
▪               SensorFlagCommandable      = no
▪               SensorFlagManageable       = no
▪               SensorDataArchiveDirectory = X
▪               SensorPropArchiveDirectory =
▪       </MCISensor>
▪       <MCISensor>
▪               SensorID                   = TTW1Dewar_TempFirstStage
▪               SensorName                 = TempFirstStage
▪               SensorUnit                 = K
▪               SensorManufacturer         = X
▪               SensorModel                = X
▪               SensorPosition             = X
▪               SensorUpdateInterval       = X
▪               SensorResolution           = X
▪               SensorDataAvailabilityTime = X
▪               SensorMinLimit             = X
▪               SensorMaxLimit             = X
▪               SensorMinWarningLimit      = 0
▪               SensorMaxWarningLimit      = 30
▪               SensorMinAlertLimit        = 0
▪               SensorMaxAlertLimit        = 32
▪               SensorFlagProvider         = yes
▪               SensorFlagConsumer         = no
▪               SensorFlagCommandable      = no
▪               SensorFlagManageable       = no
▪               SensorDataArchiveDirectory = X
▪               SensorPropArchiveDirectory =
▪       </MCISensor>
▪       <MCISensor>
▪               SensorID                   = TTW1Dewar_TempSecondStage
▪               SensorName                 = TempSecondStage
▪               SensorUnit                 = K
▪               SensorManufacturer         = X
▪               SensorModel                = X
▪               SensorPosition             = X
▪               SensorUpdateInterval       = X
▪               SensorResolution           = X
▪               SensorDataAvailabilityTime = X
▪               SensorMinLimit             = X
▪               SensorMaxLimit             = X
▪               SensorMinWarningLimit      = 0
▪               SensorMaxWarningLimit      = 10
▪               SensorMinAlertLimit        = 0
▪               SensorMaxAlertLimit        = 12
▪               SensorFlagProvider         = yes
▪               SensorFlagConsumer         = no
▪               SensorFlagCommandable      = no
```

```
▪             SensorFlagManageable       = no
▪             SensorDataArchiveDirectory = X
▪             SensorPropArchiveDirectory =
▪     </MCISensor>
▪     <MCISensor>
▪             SensorID                   = TTW1Dewar_Pressure
▪             SensorName                 = Pressure
▪             SensorUnit                 = e-4 mbar
▪             SensorManufacturer         = X
▪             SensorModel                = X
▪             SensorPosition             = X
▪             SensorUpdateInterval       = X
▪             SensorResolution           = X
▪             SensorDataAvailabilityTime = X
▪             SensorMinLimit             = X
▪             SensorMaxLimit             = X
▪             SensorMinWarningLimit      = 0
▪             SensorMaxWarningLimit      = 4
▪             SensorMinAlertLimit        = 0
▪             SensorMaxAlertLimit        = 10
▪             SensorFlagProvider         = yes
▪             SensorFlagConsumer         = no
▪             SensorFlagCommandable      = no
▪             SensorFlagManageable       = no
▪             SensorDataArchiveDirectory = X
▪             SensorPropArchiveDirectory =
▪     </MCISensor>
▪ </MCISensorControlPoint>
▪
```

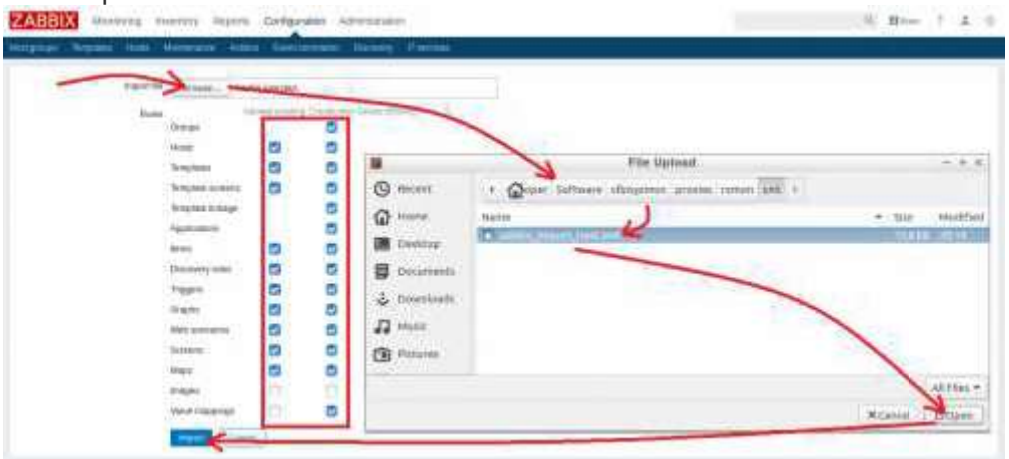
- Register the new sensor control point:
  - The registration is processed using the function "*CSysMonAPI.usRegisterSensors(argv[1])*" in the above program
  - Another possibility to register the sensors is to use the "sysmon_senderc" program, which can be found here "*/home/oper/Software/vlbisysmon/main/sysmon_sender/bin/* " in the described installation. Call the program with the following parameter:

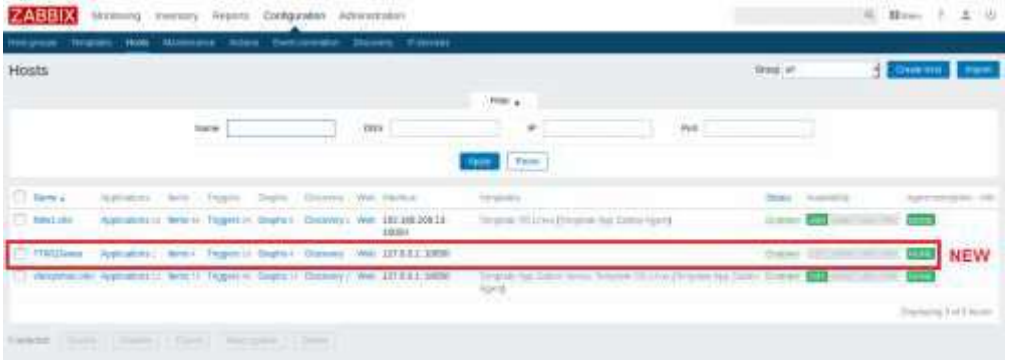
```
▪      sysmon_senderc -R dewarproxyc.conf
▪
```

4.5.3 Import sensor control point template as host to Zabbix

- The registration creates a separate XML-file if the configuration file contains a valid "*<CreateZabbixImportTemplates*" block.
- The generated XML-file describes the complete host settings for the new sensor control point and can directly be imported to Zabbix. The import can be done using the web interface of Zabbix.
- Start the import using "**Configuration→Hosts→Import**"

- 

- Select the rules for “Update existing”, “Create new” and “Delete missing” an browse to your file in the “File Upload” window. Select the newly create XML template file, click “Open” and then “Import”.
- 

- After the import you should find a new host with some items, triggers, graphs, etc. in the list of hosts
- 

4.5.4 Send in data and check the arrival

- There are two possibilities to send data:
  - Start your C or C++ program (or proxy)

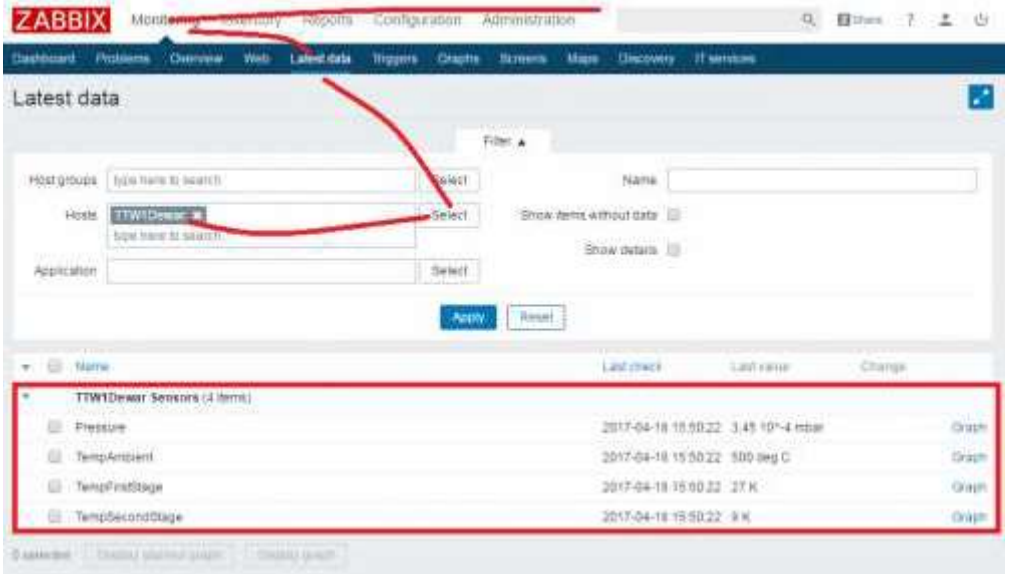
```
** Open connection to rxmon server at 192.168.208.13:52666
** Read dewar values from rxmon server at 192.168.208.13:52666
500deg C
info from server: "processed: 1; failed: 0; total: 1; seconds spent: 0.000047"
sent: 1; skipped: 0; total: 1
27.000K
info from server: "processed: 1; failed: 0; total: 1; seconds spent: 0.000044"
sent: 1; skipped: 0; total: 1
9.000K
info from server: "processed: 1; failed: 0; total: 1; seconds spent: 0.000028"
sent: 1; skipped: 0; total: 1
1.591000010^-4 mbar
info from server: "processed: 1; failed: 0; total: 1; seconds spent: 0.000051"
sent: 1; skipped: 0; total: 1
** Close connection to rxmon server.
```

  - or run the program “sysmon_senderc” in the directory , which can be found here “*/home/oper/Software/vlbisysmon/main/sysmon_sender/bin/* ” using:

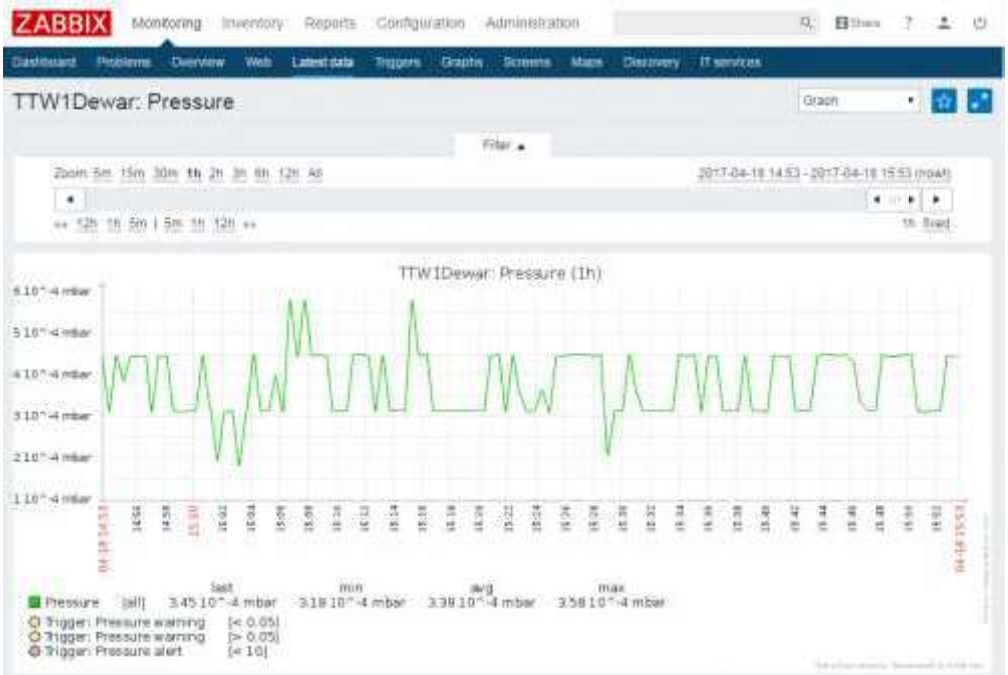
```
sysmon_senderc -s dewarproxyc.conf TTW1Dewar_TempAmbient 23.0
```

```
▪
▪     Arguments:
▪     ==========
▪     -s dewarproxyc.conf   => the configuration file used
▪     TTW1Dewar_TempAmbient => SensorID
▪     23.0                  => New value
▪
```

- Check if the data arrive in Zabbix.



-
- You can directly click on “Graph” link behind each received new value to show the data history of the already received data.



-
- Administrators can also check if the data arrive in the sysmon databases using “psql” program of PostgreSQL.

```
▪     psql -h 127.0.0.1 -p 5432 sysmon sysmon
▪     select * from mcicurrentvalues;
▪
```

- You should see something like this for your values:

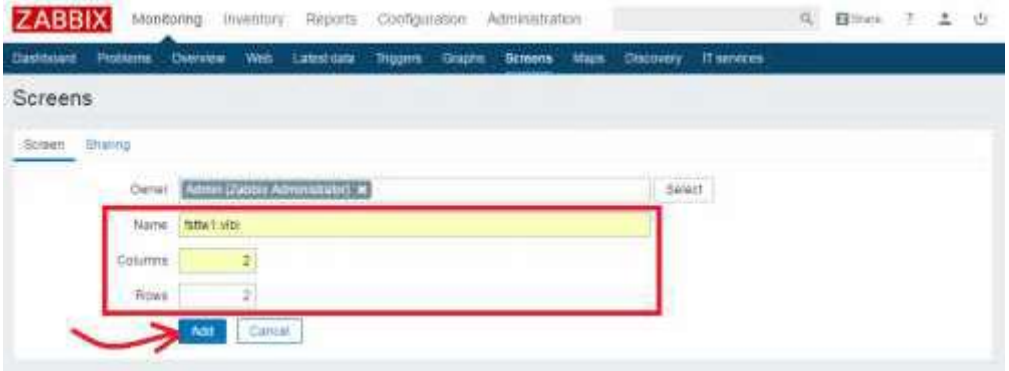
```
sysmon=# select * from mcicurrentvalues ;
        netsensorid          |       mjd       | alarmlevel | value
-----------------------------+-----------------+------------+-------
 TTW1Dewar_TempAmbient       | 57861.58190972  |          3 |   500
 TTW1Dewar_TempFirstStage    | 57861.58190972  |          3 |    27
 TTW1Dewar_TempSecondStage   | 57861.58190972  |          3 |     9
 TTW1Dewar_Pressure          | 57861.58190972  |          3 | 3.446
(4 rows)
```

- End with Ctrl-Shift-'D'

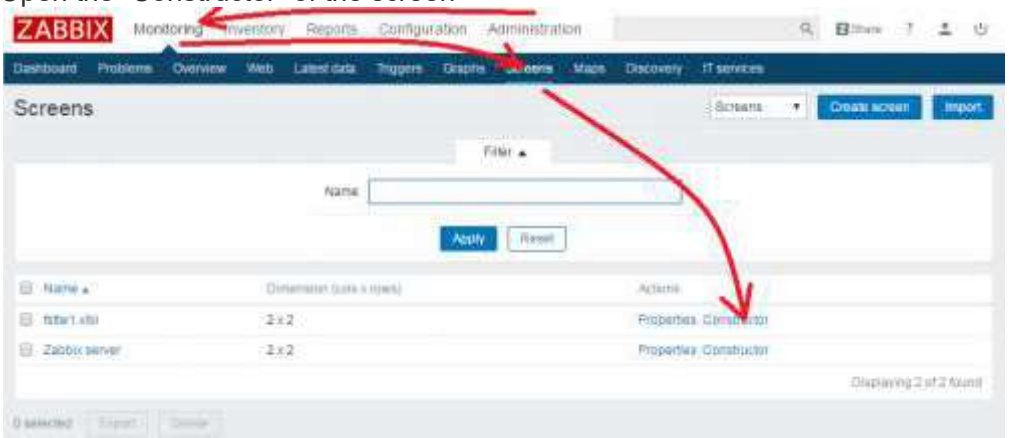
4.5.5 Create individual screens and maps

- Create a new screen using "**Monitoring→Screens→Creat screen**"

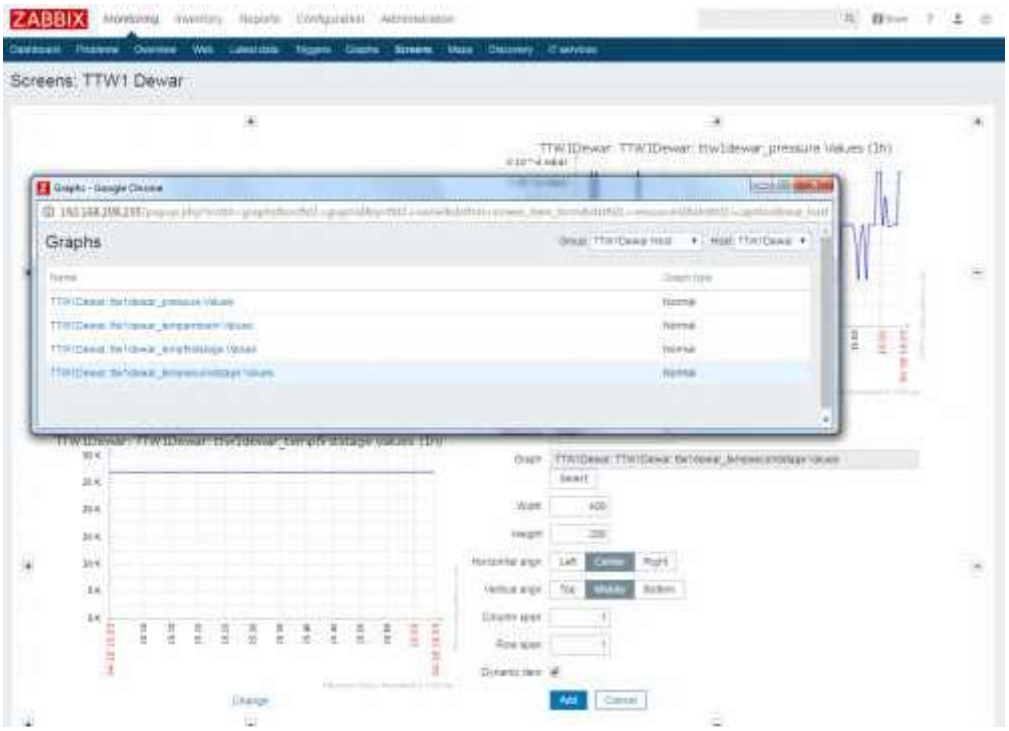
- Define a name for the screen and the dimensions as number of rows and columns and push "Add".

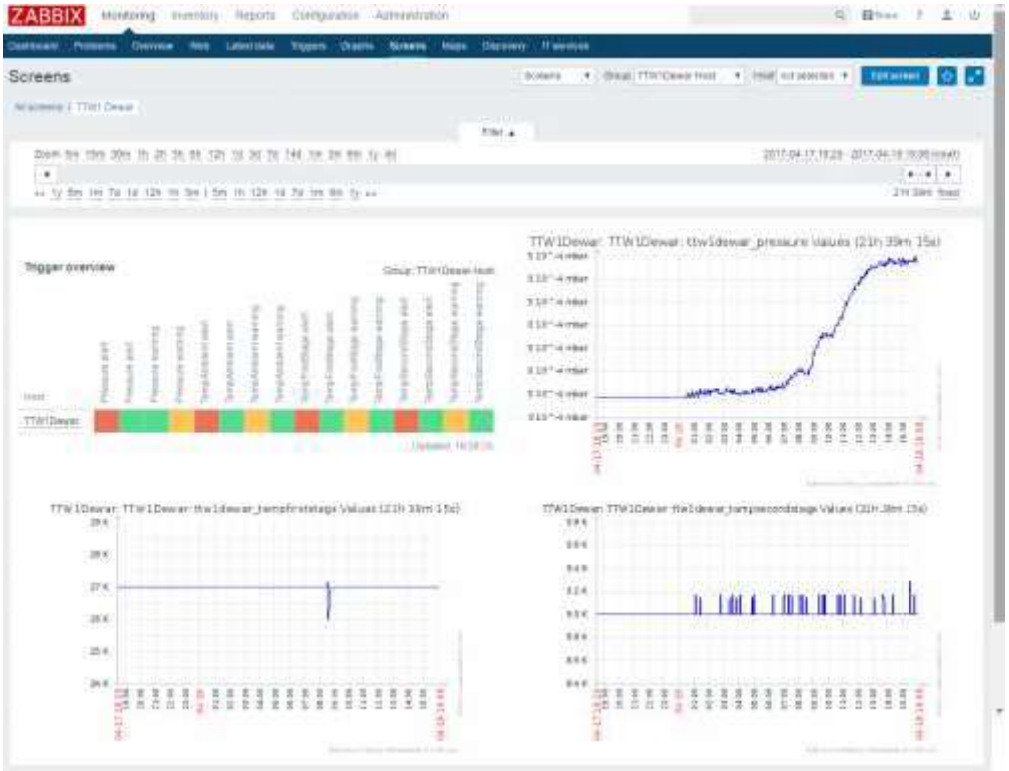


- Open the "Constructor" of the screen



- Push the "Change" link for each of the individual fields on the screen.

- 

- Do this for all individual graphs and elements which should be shown on the screen.
- The finished screen should then look like this
- 

4.6 Appendix: Installation and configuration of the monitoring of a NASA FS using e-RemoteCtrl web application

This is just a short description which will be extended for the final setup.

Download and install e-RemoteCtrl.

Configure the web server with the e-RemoteCtrl configuration file “eremotectrl.conf”:

```
    <WebServer>
        Activated = yes             # Run as web server (yes/no)
        ProtocolType = HTTP         # Currently only HTTP is possible
                                    # (in future maybe HTTPS)
        MonitoringPort = 8080       # Used port for web server to request
                                    # monitoring data
        CommandingPort = 8081       # Used port for web server to request
                                    # monitoring data and send commanding orders
        PortReuseTimeout = 10       # Timeout which is used to latest reuse
                            port after close
        HTMLTemplateDirectory = /usr2/st/html/
                                    # Directory with all HTML files as templates
        HTMLWebcamAddress = http://xyz.de/record/current.jpg
                                    # Directory with all HTML files as templates
        HTMLRefreshTimeMillisec = 1000          # Refresh time of HTML web pages
        HTMLWebCamRefreshTimeMillisec = 500     # Refresh time of HTML web cam
                                                # image
        HTMLLogRefreshTimeMillisec = 10000      # Refresh time of HTML log text
</WebServer>
```

Start the e-RemoteCtrl server “ercd eremotectrl.conf”. Now the web pages should be accessible. You can adapt the web pages to install other designs.

Request an SSH key for the access to the vlbisysmon.evlbi.wettzell.de machine. Install a “autossh” connection to the machine and inform the monitoring server operator to activate the monitoring of the new antenna.

After you got access data, you can login to the monitoring web page for your antenna.

4.7 Appendix: Installation and configuration of Grafana in addition to ZABBIX

Written by J. Bachem for the SLR system

The Grafana framework will be used to get a better visual representation of the items monitored with Zabbix. Up to date installation guidelines for Ubuntu can be found here.

```
wget https://s3-us-west-2.amazonaws.com/grafana-
releases/release/grafana_4.6.3_amd64.deb
sudo apt-get install -y adduser libfontconfig
sudo dpkg -i grafana_4.6.3_amd64.deb
sudo update-rc.d grafana-server defaults
sudo systemctl enable grafana-server.service
```

Unfortunately Grafana has it's own Web-Server running on port :3000, but we want to access the GrafanaWeb-pages in a subfolder on the standard http-port :80. To enable this we need to setup Apache to act as a proxy for the Grafana server.
First we need to enable the Apache proxy modules:

```
sudo a2enmod proxy
sudo a2enmod proxy_html
sudo a2enmod proxy_http
```

Then modify the default Web-Server settings in ”/etc/apache2/sites-available/000-default.conf“ and add this settings:

```
# Settings to proxy the Grafana server
    ProxyRequests Off
    <Proxy *>
          Order deny,allow
          Allow from all
    </Proxy>
    ProxyPass /grafana/ http://127.0.0.1:3000/
    ProxyPassReverse /grafana/ http://127.0.0.1:3000/
```

and add/change the following setting in ”/etc/grafana/grafana.ini“:

```
root_url = http://localhost:3000/grafana
```

Finally we need to install the Grafana-plugin for Zabbix:

```
grafana-cli plugins install alexanderzobnin-zabbix-app
```

and restart both Apache and Grafana:

```
sudo service grafana-server restart
sudo service apache2 restart
```

Next steps are to configure the Grafana Zabbix plugin:

1. Login to the Grafan Web page (http://wlrs-s1.wlrs/grafana/) with user “admin” and password “admin”
2. In the top-left menu open “Plugins”
3. Click the Apps tabs in the Plugins section and select the Zabbix plugin
4. Click on enable
5. In the top-left menu select “Data sources” and add a new data source with the following settings
   a. Name: Zabbix
   b. Type: Zabbix
   c. URL: http://127.0.0.1:80/zabbix/api_jsonrpc.php

d. Access: Proxy
e. Username: Admin
f. Password: Zabbix

6. Click on “Save and Test”

Now the Zabbix Server Dashboard within the top-left menu → Zabbix should be working.

Installation of TIG monitoring system for vlbi_nodes and integration of Zabbix

Barbieri Edoardo – edoardo.barbieri@tum.de

The following installation has been tested on Ubuntu desktop 16.04 LTS, which is also recommended.
The installation includes four software components:

- **Telegraf** (for vlbi): it is the data collector. It should be installed on every machine/node being monitored.
- **InfluxDB**: it is a time series database. It collects data from all the nodes on which Telegraf is running. Uses a **push model** → clients initiate the connection and write to the database.
- **Grafana**: the graphical interface of the system. It runs a web interface on which is possible to query the database and create customizable dashboards.
- **Zabbix plugin for Grafana:** a Grafana’s plugin that allows us to import metrics from Zabbix system.

Recommended configuration of the system is InfluxDB+Grafana on the server used for monitoring, Telegraf on the machines/nodes to monitor.

Installation of InfluxDB and Grafana

To proceed with the following steps we need to be root:

```
sudo -i
```

In order to download the software packages, we first need to import the keys from dedicated repositories

```
curl -sL https://repos.influxdata.com/influxdb.key | apt-key add -
curl -sL https://packagecloud.io/gpg.key | apt-key add -
```

You should get an “OK” if both operations succeeded.
Now we have to create a file to add the repositories to package manager with any text editor. Here we use *nano*.

```
nano /etc/apt/sources.list.d/tig.list
```

And we copy/past the following for grafana

```
#####################
## Grafana repo
## Use for all Debian/Ubuntu variants
```

```
deb https://packagecloud.io/grafana/stable/debian/ jessie main
```

And for InfluxDB we need to know the Ubuntu/debian variant we are using. This can be done through the command:

```
source /etc/os-release && echo $VERSION
```

Then we can copy/past the following but uncommenting only the appropriate line and, in case we use Ubuntu, replacing *xenial* with the right version

```
####################
## InfluxData repo
## Uncomment the appropriate line
## Wheezy
#deb https://repos.influxdata.com/debian wheezy stable
#
## Jessie
#deb https://repos.influxdata.com/debian jessie stable
#
## For Ubuntu, replace xenial with appropriate codename
#
#deb https://repos.influxdata.com/ubuntu xenial stable
```

Then we can save the file (ctrl^O + ctrl^x with *nano*).
Now we update the package manager's database, and install the software

```
apt-get update

apt-get install influxdb
apt-get install grafana
```

Now to enable Grafana starting on boot:

```
systemctl daemon-reload
systemctl enable grafana-server

#And to start the servers
systemctl start grafana-server
service influxdb start
```

Now you should be able to access grafana by typing in your browser http://localhost:3000 while influxdb is working at the port 8086, but without displaying any web page.

Installation of Telegraf

To install the standard version of Telegraf (monitoring of cpu, memory and disk usage), it is already available from InfluxDB repository, and you only have to run

```
apt-get install telegraf
```

To install the VLBI branch of telegraf you need to have access to the following repos
https://user:pass@lupus.gsfc.nasa.gov/fs/debian
where you replace user and pass with your credential.
If you do have access, add the Field System repos by creating the file

```
nano /etc/apt/sources.list.d/lupus.list
```

Copy/paste the following repositories (replacing the credentials)

```
deb https://user:pass@lupus.gsfc.nasa.gov/fs/debian wheezy main
```

Then get the GPG key

```
apt-key adv --keyserver keys.gnupg.net --recv-keys 6E2CE741
```

Finally update & install

```
apt-get update
apt-get install telegraf-vlbi
```

Now you should have Telegraf installed. To launch telegraf

```
service telegraf start
```

Configuration of InfluxDB

InfluxDB configuration file is located in */etc/influxdb/influxdb.conf*
For information about InfluxDB configuration you can find the documentation at

https://docs.influxdata.com/influxdb/v1.2/administration/config/

Configuration of Grafana

Grafana configuration file is located at */etc/grafana/grafana.ini*
Find full documentation at
http://docs.grafana.org/installation/configuration/

Configuration of Telegraf

Telegraf configuration file is located at */etc/telegraf/telegraf.conf*
Modify the .conf file as follows

```
17   # Global tags can be specified here in key="value" format.
18   [global_tags]
19     # dc = "us-east-1" # will tag all metrics with dc=us-east-1
20     # rack = "1a"
21     ## Environment variables can be used as tags, and throughout the config file
22     # user = "$USER"
23   station="gs"
24
```

...
In the [agent] tag you can modify the field "interval" and "flush interval" and other parameters if necessary. This depends on the upload rate at which we want the data. Description of each parameter is provided in the same file. Otherwise leave it as default (10 sec).

...

```
###############################################################################
#                            OUTPUT PLUGINS                                   #
###############################################################################

# Configuration for influxdb server to send metrics to
[[outputs.influxdb]]
  ## The full HTTP or UDP URL for your InfluxDB instance.
  ##
  ## Multiple urls can be specified as part of the same cluster,
  ## this means that only ONE of the urls will be written to each interval.
  # urls = ["udp://127.0.0.1:8089"] # UDP endpoint example
  urls = ["http://localhost:8086"] # required
  ## The target database for metrics (telegraf will create it if not exists).
  database = "vlbi" # required
```

Replace urls field with address of influxDB server (where is located your database).
Change name of database into "vlbi".

Here you can also allow other different outputs for telegram by uncommenting them.

...

```
###############################################################################
#                            INPUT PLUGINS                                    #
###############################################################################

# Read metrics about cpu usage
[[inputs.cpu]]
  ## Whether to report per-cpu stats or not
  percpu = true
  ## Whether to report total system cpu stats or not
  totalcpu = true
  ## If true, collect raw CPU time metrics.
  collect_cpu_time = false
  ## If true, compute and report the sum of all non-idle CP
  report_active = false
```

Under the [input] tags is possible to select which component telegraf should monitor and collect data from. By default system components (i.e. cpu, memory, disk, etc.) are selected. To monitor a new data source it's necessary to uncomment the relative part in the file.
To add new measurement inputs you need to install the relative plugin. See more at
https://docs.influxdata.com/telegraf/v1.2/administration/configuration/

...

After changing the configuration it is necessary to restart telegraf and influxdb.

```
service telegraf restart
service influxdb restart
```

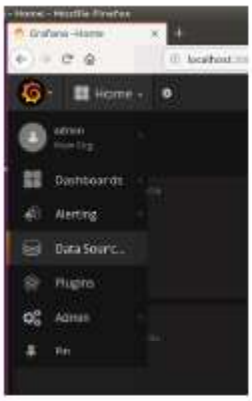
Getting started with Grafana

Grafana server should be set up to start on boot. If it doesn't follow the instruction in the previous section.
Once it's done it must be possible to access Grafana from any browser at
http://<serveraddress>:3000

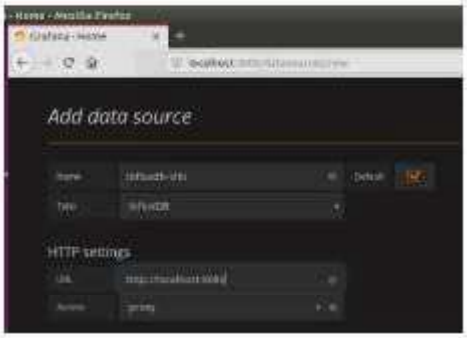
First access credentials:
User : *admin*
Password: *admin*

Then to add the database:

Go to "Data Source" → Click on "Add data source"



Then copy the following configuration



Leave the fields "User" and "Password" blank in case you haven't configured Influxdb otherwise. And finally click on "Add" button.

If you did everything correctly you should get the following message

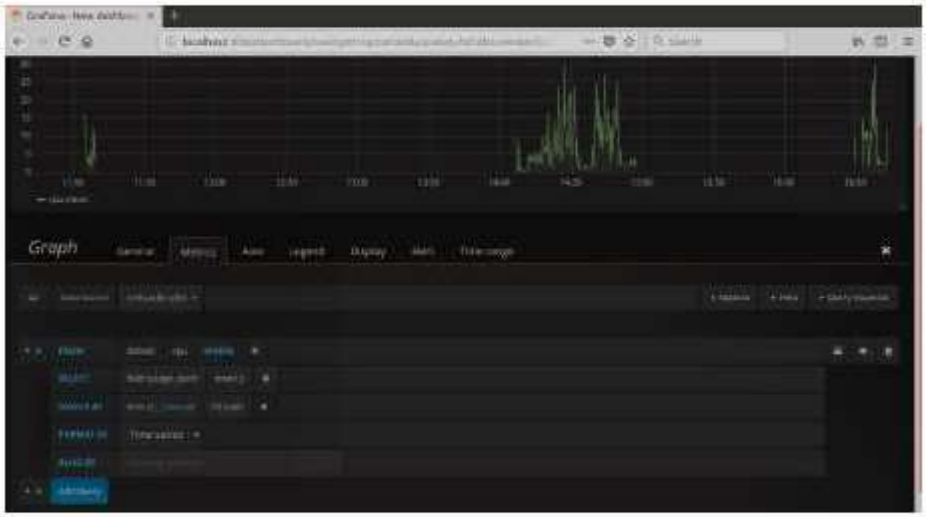
Important: do not forget to change the credentials (user/password),

Now we are able to send *queries* to the database through Grafana and customize the dashboard with them.

Choose a panel and Drag&drop to the dashboard. Eg. "graph panel"

Click on "panel title → edit" to edit the panel.

Under the "metrics" tab we can form our *query* selecting the telegraf database and this way test the connection.



If everything was configured correctly we can see the objects stored in the database through a *dropdown* list by clicking on each field. As we select an object, data will be automatically displayed in the graph.

Please refer to Grafana documentation if you want to exploit Grafana's features.
http://docs.grafana.org/

Integration of Zabbix and Grafana

This requires to have a Zabbix system already installed and running on your server.

Visit the following website and search for Zabbix plugin
https://grafana.com/plugins?utm_source=grafana_plugin_list

Here the instruction for the installation and docs are given, but to make it short you only need to run the following


```
sudo grafana-cli plugins install alexanderzobnin-zabbix-app
```

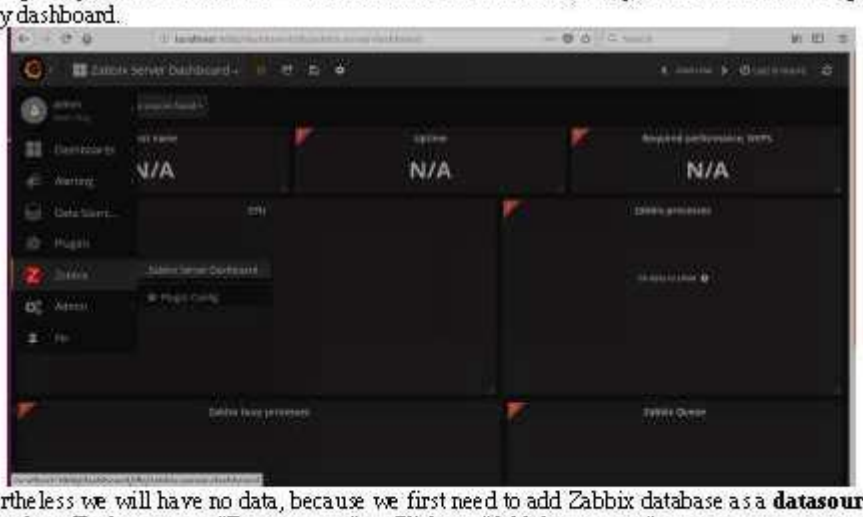
And restart Grafana

```
sudo service grafana-server restart
```

After doing this the plugin will be available in your Grafana under "Plugins→Apps"

Click on the icon of the App and then click on "Enable".

At this point you should have a Zabbix icon in the cascade menu, and you should be able to open an empty dashboard.



Nevertheless we will have no data, because we first need to add Zabbix database as a **datasource** for Grafana. To do so go to "Data sources" → Click on "Add data source".

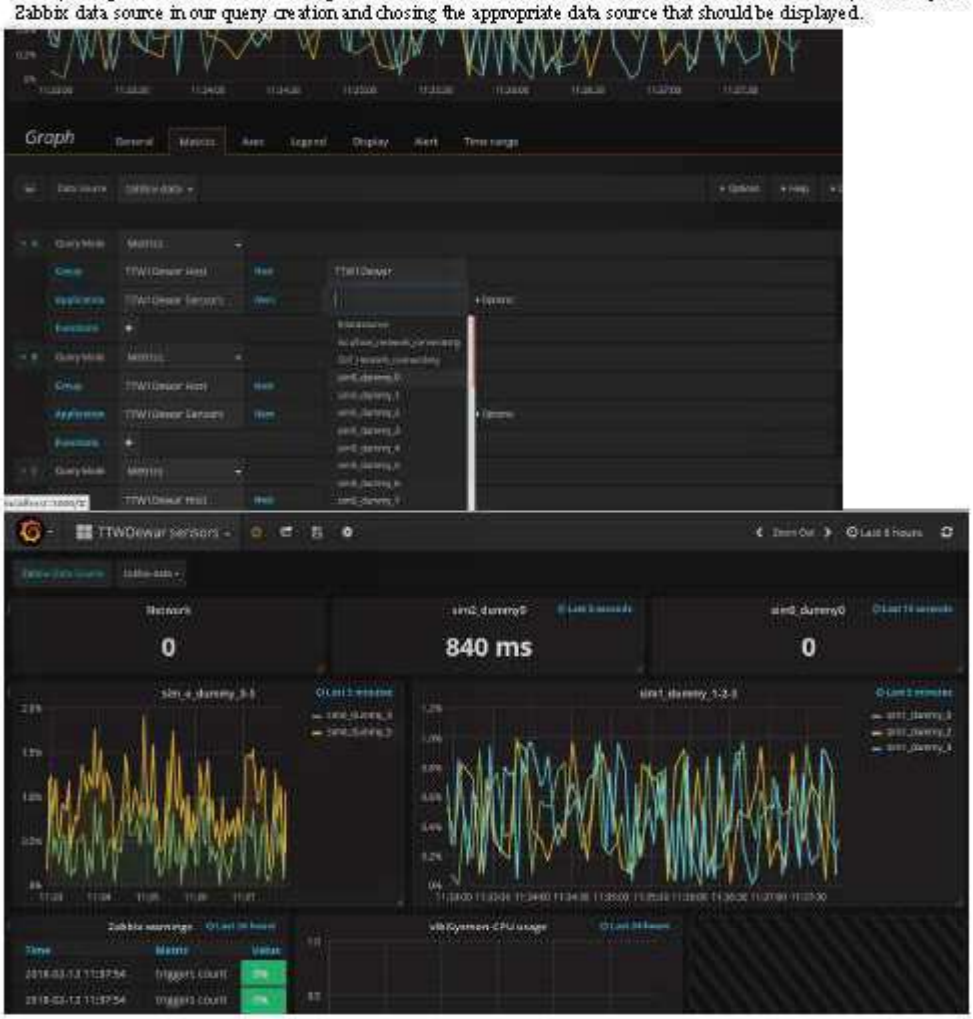
Copy the following configuration

The URL used to fetch zabbix data is as follows

http://<zabbix-server-address>/api_jsonrpc.php

The credentials are the same as zabbix user credentials (the one used to login into zabbix)

Finally it is possible to visualize and explore our Zabbix data sources and create customized dashboard by selecting the Zabbix data source in our query creation and chosing the appropriate data source that should be displayed.



Additional Documentation

InfluxDB
https://docs.influxdata.com/influxdb/v1.1/introduction/installation/
https://docs.influxdata.com/influxdb/v1.2/administration/config/
Telegraf
https://docs.influxdata.com/telegraf/v1.2/administration/configuration/
Grafana
http://docs.grafana.org/

References

David Horsley (david.e.horsley@nasa.gov)– TIGf or VLEI Operations pdf