Symbiosis of smart objects across IoT environments

symbIoTe - H2020-ICT-2015

Platform Monitoring component

The symbIoTe Consortium

Intracom SA Telecom Solutions, ICOM, Greece
Sveučilisteu ZagrebuFakultetelektrotehnikeiračunarstva, UNIZG-FER, Croatia
AIT Austrian Institute of Technology GmbH, AIT, Austria
NextworksSrl, NXW, Italy
ConsorzioNazionale Interuniversitario per le Telecomunicazioni, CNIT, Italy
ATOS Spain SA, ATOS, Spain
University of Vienna, Faculty of Computer Science, UNIVIE, Austria
UnidataS.p.A., UNIDATA, Italy
Sensing & Control System S.L., S&C, Spain
Fraunhofer IOSB, IOSB, Germany
Ubiwhere, Lda, UW, Portugal
VIPnet, d.o.o, VIP, Croatia
InstytutChemiiBioorganicznejPolskiejAkademiiNauk, PSNC, Poland
NA.VI.GO. SCARL, NAVIGO, Italy
ATOS Wordline International, WLI, Spain

For more information on this document, please contact:
David Rojo, ATOS Spain, david.rojoa@atos.net
Fernando Campos, ATOS Spain, fernando.camposdiaz@atos.net

Document Control

Title: Platform Monitoring component

Type: Public

Editor(s): David Rojo and Fernando Campos, ATOS

E-mail: david.rojoa@atos.net; fernando.camposdiaz@atos.net

Author(s): David Rojo and Fernando Campos (ATOS)

Doc ID: platform-monitoring-atos.docx

Amendment History

Version	Date	Author	Description/Comments
V0.1	April13th	David Rojo	Initial ToC and first version of sections 1 to 5
V0.2	April 18th	Fernando Campos	First revision
V0.3	April 21th	David Rojo	Reviewed by Vasilis Glykantzis (ICOM), Core Resource Monitor is used instead of Core Resource Access Monitor to send Monitoring information. Modified by David Rojo
V0.4	May 11th	David Rojo	Changes added to params sent to check_symbiote_iot.sh, now id and type are sent

Legal Notices

The information in this document is subject to change without notice.

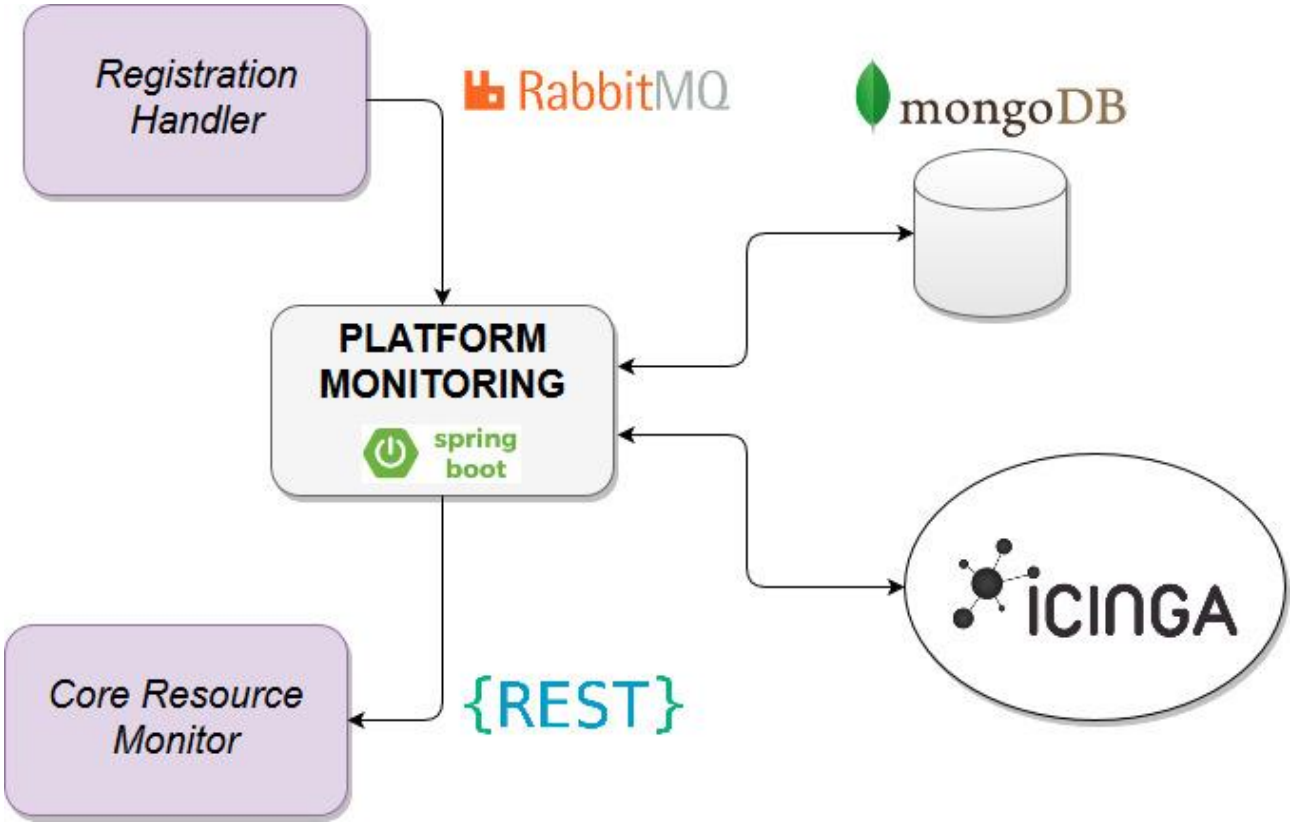
The Members of the symbIoTe Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the symbIoTeConsortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

(This page is left blank intentionally.)

Table of Contents

1 Platform Monitoring

1.1 Architecture



Platform Monitoring is a component deployed in every platform that belongs to SymbIoTe. It has been developed using Spring Boot framework and it is composed by a mongoDB database where the devices of the platform are stored. Also a very important element of the component is icinga2, an open source tool for monitoring resources, that in this case is used to monitor the devices of the platform.

1.2 Interfaces

The component use two ways of communication using Rabbit MQ for Registration Handler and REST for Core Resource Monitor.

1.2.1 Registration Handler

Registration Handler send information to Platform Monitoring using Rabbit MQ, three operations are defined:

- Add devices: a list of devices to add to the component. More details in the 4.2 section of this document.
  - Exchange name: *symbIoTe.rh.reg*
  - Queue name: *symbIoTe.monitoring.registrationHandler.register_resources*
- Delete devices: a list of devicesId to delete of the component.

  - Exchange name: *symbIoTe.rh.unreg*
  - Queue name: *symbIoTe.monitoring.registrationHandler.unregister_resources*
- Update devices: a list of devices to update
  - Exchange name: *symbIoTe.rh.update*
  - Queue name: *symbIoTe.monitoring.registrationHandler.update_resources*

The queues are declared with the following configuration:

- durable = true
- autoDelete = false
- exclusive = false

1.2.2 Core Resource Monitor

Platform Monitoring sends monitoring information per platform periodically using REST to Core Resource Monitor.

The endpoint where the information is sent is *crm/monitoring/{platformId}/devices/status*

Is possible to modify the period time that the information is published if the value for the property symbiote.crm.publish.period is modified in the properties file of the component. The expression must be in cron format, by default is every minute (0 * * * * ?).

The message that is sent to the endpoint is an instance of CloudMonitoringPlatform that contains:

- internalId: the platform id of the platform. String
- coreToken: an instance of SymbIoTeToken class provided by Security Handler
- devices: an array of devices where every position is a device existing in the platform and contains:
  - id: symbiote id of the device.String
  - availability: int number that define if the device is available (1) or not (0)
  - load: int number that define the load of the device. If a device is not available its load is -1. If the device is available the load is a number between 0 and 100.
  - timestamp: date of the measurement. String

2 Making a platform compliance with Symblote Platform Monitoring

2.1 Install the icinga2mastersoftware.

This task is fully described step by step in section5.1 of this document, please complete it before continuing

2.2 Install additional software and properties file

Is mandatory to have a **mongoDB** server installed and running on the machine that is going to be used as platform.

Also is mandatory to have the jar file of the component and have the properties file of the component, named bootstrap.properties, filled out with the right information (between <> the information that can be replaced by right values):

```
spring.application.name=Monitoring
spring.cloud.config.uri=http://localhost:8888
logging.file=logs/Monitoring.log

#if core available true
platformmon.init.autoregister=false

# rabbitmq
symbiote.rabbitmq.host.ip=<rabbit_mq_server_ip>
server.port=8200

# icingaconf
symbiote.icinga2.user.agent="symbIoTe-Platform-Monitoring-Icinga2-REST_Client/0.1"
symbiote.icinga2.api.url=https://<icinga_server_hostname>:5665/v1
symbiote.icinga2.api.user=<icinga_api_user>
symbiote.icinga2.api.password=<icinga_api_password>

# every minute monitoring info is published to CRM
symbiote.crm.publish.period=0 * * * * ?
symbiote.crm.url=http://<crm_ip>:<crm_port>

# platform ID
platform.id=<platform_id>

# Security handler
symbiote.sh.user=<security_handler_user>
symbiote.sh.password=<security_handler_password>
symbiote.coreaam.url=http://<core_aam_ip>:<port>
```

3 Adding a new host to existing SymbIoTe platform

3.1 Install icinga2 satellite software

This task is fully described step by step in section 5.2 of this document, please complete it before continuing

NOTE: Adding new host is not possible, if the icinga2 master is not installed.Also no new devices can be added to the host, if the host does not exist in icinga2.

4 Adding devices to existing host in existing symbiote platform

When a host is registered in icinga2 environment, is possible to add new devices (services in icinga2 terminology) to it but after of this, is mandatory to complete some requirements previously. After doing these, the RabbitMQ communication between Registration Handler and the Platform Monitoring can be used.

4.1 Requirements

Platform Monitoring register the load and the availability of every device, these values are provided by the execution of a script named `check_symbiote_iot.sh`, that must be allocated in /usr/lib/nagios/plugins path and must has permission of execution:

```
root@client:~# chmod 755 check_symbiote_iot.sh
```

When the script is executed, it receives 2 values:

- internalid: the internal id of the device
- type: the type of the device

NOTE:these parameters have been selected by the developers of the component, but if is needed to be changed, please contact with them. At least, the internal id must be sent to the script in order to be able to distinguish

Exists one file of this kind in every platform, and if more than one device exists on the platform, is needed to take account the internal id of each device and depending of this, execute different commands in order to provide the right load and the right availability for each device.

When a device is available the value for availability is 1, and if not is 0

When a device is not available (availability=0), the value for load is -1, if it is available (availability=1) the value for load is a number between 0 and 100.

The output of the script must have the following format with no more text:

```
symbiote result on
client#AVAILABILITY=<AVAILABILITY_VALUE>#LOAD=<LOAD_VALUE>
```

If the script does not provide an output with this format, Platform Monitoring component will not be able to retrieve the values for availability and load.

Below is provided an example of check_symbiote_iot.sh that can be used as starting point.

```
#!/bin/bash


# extract the values of the params and store them in many vars
whilegetopts s:d:i: option
do
case "${option}"
```

```
in      s) IOT_INTERNALID=${OPTARG};;
t) IOT_DEVICE_TYPE=${OPTARG};;
esac
done

# generate random numbers between 0 and 1
# 0 = device is not available
# 1 = device is available
availability=$((RANDOM%2))
if [ $availability -eq 0 ]
then
        # if the device is not available, the load value is -1 that means
that there is not information about it
load=-1
else
        # if the device is available, the load value is a value between 0
and 100
load=$(((RANDOM%101)-1))
fi
# provide the output with the right format
echosymbiote result on client#AVAILABILITY=$availability#LOAD=$load

exit 0
```

4.2 Adding devices

When the requirements has been completed is possible to register new devices in the host, for this is used the RabbitMQ communication with Registration Handler, that send messages to Platform Monitoring.

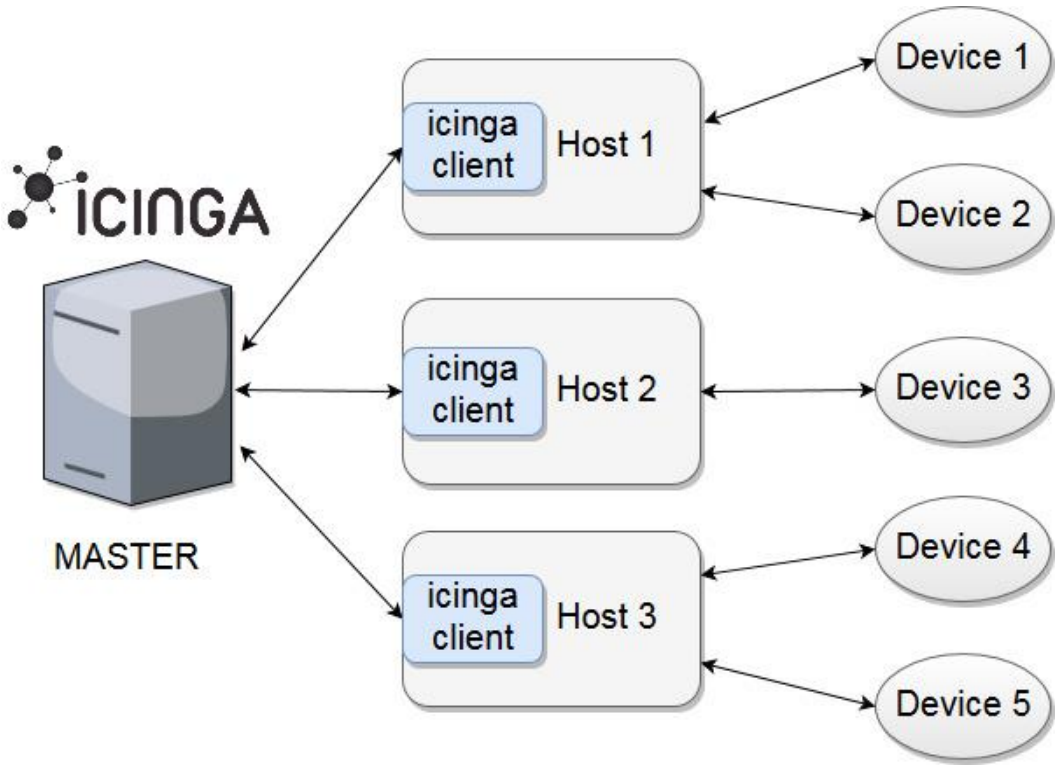
The content of these messages is a list of instances of CloudResource class that contains:

- internalId: the internal id of the device. Must be unique
- host: the ip address of the host where the device is. The host must exists in icinga2
- resource: an instance of the Resource class
- params: an instance of the CloudResourceParams class that contains:
    - type: the type of the device

NOTE: CloudResourceParams class is used to send data to icinga2 script, now information sent is type and internalId (extracted from CloudResource)

5 Icinga2

Icinga2 is an open source monitoring system that is able to check the availability of your network resources and services. Also allow notification to users and generate performance data for reporting tasks. It was originally created as a fork of another monitoring system, Nagios, but adding new features like a modern Web 2.0 style user interface, additional database connectors (for MySQL/MariaDB, Oracle and PostgreSQL) and a REST API that allow the developers the creation of new extensions.



From Platform Monitoring point of view, every platform must have an Icinga2 master running (steps described in 5.1 section of this document). Also every host that belongs to the platform, must have installed the Icinga2 client (steps described in 5.2 section of this document) in order that the devices of every host of the platform can be monitored in Icinga2 master.

5.1 Installing Icinga2 master

In this section is listed the steps thar are needed to set up an Icinga2 server with Web 2 interface on Ubuntu server.

5.1.1 Adding repositories

Icinga2 repositories must be added to the server. Is needed to enable the add-repository feature, then add its repositories to the repository packages, and update the packages.

```
root@master:~# apt install software-properties-common
root@master:~# add-apt-repository ppa:formorer/icinga
root@master:~#apt-get update
```

5.1.2 Installing Icinga2

With the repositories added, Icinga2 can be installed:

```
root@master:~# apt-get install icinga2
```

When the installation finish, be sure that the service is up and working fine:

```
root@master:~# systemctl enable icinga2.service
root@master:~# systemctl start icinga2.service
```

5.1.3 Enabling the Feature list

Icinga2 provides many features that can be enabled or disabled depending of the needs of the user. The features that are enabled by default are shown using the command:

```
root@master:~# icinga2 feature list
```

The features enabled by default are:

- Checker: enables the execution of checks
- Mainlog: enables the logging
- Notification: enables notification mechanism

For the usage of this tool in SymbIoTe project, the API feature must be enabled in order to communicate the Platform Monitoring component with the Icinga2 master node. For this, execution of this command is needed:

```
root@master:~# icinga2 api setup
```

After of this, icinga2 must be restarted manually:

```
root@master:~# service icinga2 restart
```

5.1.4 Install Icinga2 plugin

Icinga2 collect the service information based on the monitoring plugin, for this task, the installation of the Nagios plugins are needed:

```
root@master:~# apt-get installnagios-plugins
```

5.1.5 Installing Data Output (IDO) Module

MySQL has been chosen as external database, so is needed the installation of the MySQL IDO module that is used by Icinga2 web interface. This module is used for exporting all status and configuration information to the database. If is not installed before, the installation of MySQL in the server is needed.

```
root@master:~# apt-get install mysql-server-5.7

root@master:~# apt-get install icinga2-ido-mysql
```

After installing the IDO module, is needed to setup the MySQL database to accept the values using this moduled. For this task, database named “icinga2” is created, with the username “symbiote” and password “symbiote123”

```
root@master:~# mysql -u root -p
mysql> Create database icinga2;
mysql> GRANT SELECT, INSERT, UPDATE, DELETE, DROP, CREATE VIEW,
INDEX, EXECUTE ON icinga2.*TO 'symbiote'@'localhost' IDENTIFIED BY
'symbiote123';

mysql> flush privileges;
```

Verify that the ido-mysql feature is enabled:

```
root@master:~# icinga2 feature enable ido-mysql
warning/cli: Feature 'ido-mysql' already enabled.
```

Restart icinga2 service:

```
root@master:~# service icinga2 restart
```

Verify that the IDO MySQL configuration file contains the correct DB credentials.

```
root@master:~# cat /etc/icinga2/features-enabled/ido-mysql.conf
```

```
root@VTSS031:/etc/icinga2# cat /etc/icinga2/features-enabled/ido-mysql.conf
/**
 * The db_ido_mysql library implements IDO functionality
 * for MySQL.
 */

library "db_ido_mysql"

object IdoMysqlConnection "ido-mysql" {
  user = "symbiote",
  password = "symbiote123",
  host = "localhost",
  database = "icinga2"
}
```

5.1.6 Installing and configuring Icinga Web2 plugin

In Ubuntu the7.0 is the default version for PHP, but there are a lot of compatibility issues for icinga2 with PHP 7.0, so is needed the installation of PHP v5.6. To do this the “ondrej/php” repository must be enabled:

```
root@master:/usr#add-apt-repositoryppa:ondrej/php
```

```
root@master:/usr#apt-getinstall php5.6-gd php5.6-json php5.6-dba
php5.6-intl php5.6-ldap php5.6-pdo-mysql php5.6-imagick php5.6-dom
```

A MySQL database is used for savings all insticts of Icinga2 Web interface. Executing the following steps the database is created and all priviliges are granted to the user on the database:

```
root@master:~#mysql -u root -p
mysql> create database icingawebdb;
mysql> GRANT SUPER ON *.* TO 'icingaweb'@'localhost' IDENTIFIED BY 'symbiote123';
mysql> GRANT SELECT, INSERT, UPDATE, DELETE, DROP, CREATE VIEW, INDEX, EXECUTE ON icingawebdb.*TO 'icingaweb'@'localhost' IDENTIFIED BY 'symbiote123';
```

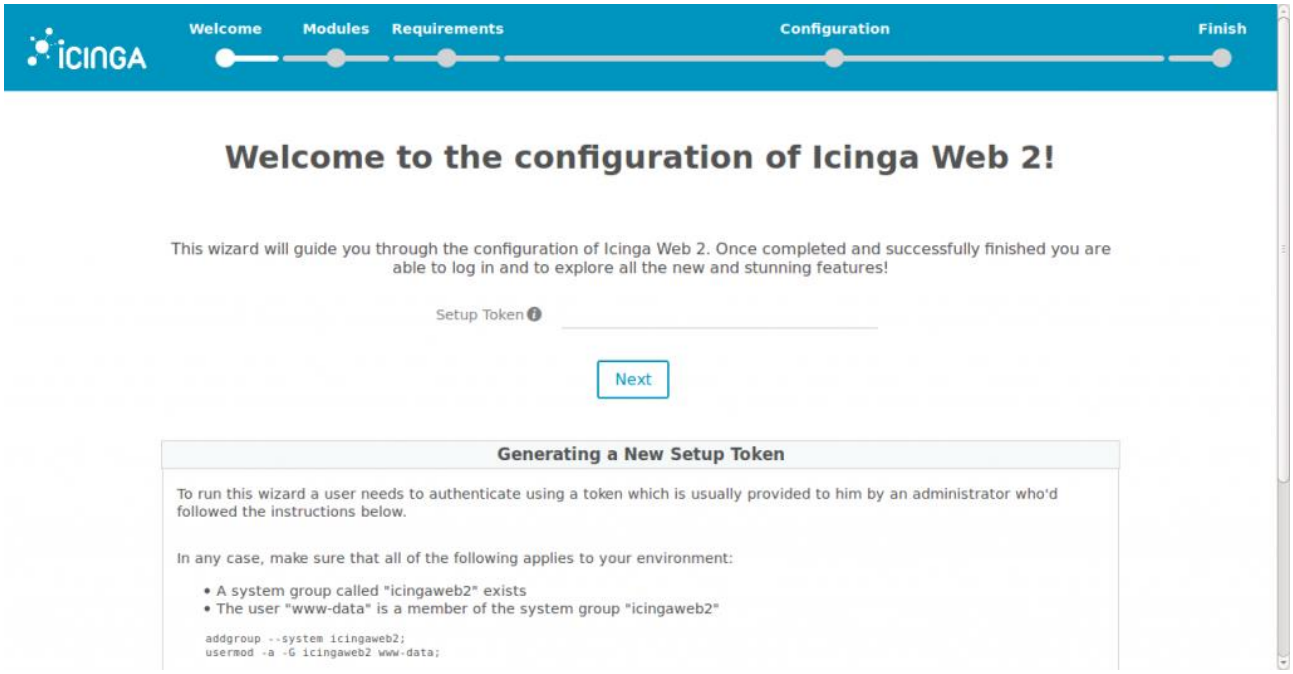
After creating the database, the installation of the Web interface plugin and the configuration of it one by one can be done.

```
root@master:~# apt-get install icingaweb2
```

When the installation finish, we can access to this URL to continue with the configuration:

```
http://<master_ip_address>/icingaweb2/setup
```

5.1.6.1 Creating the token



The initial setup page shows a message to authenticate our Icingaweb user and create a token to proceed with the configuration.

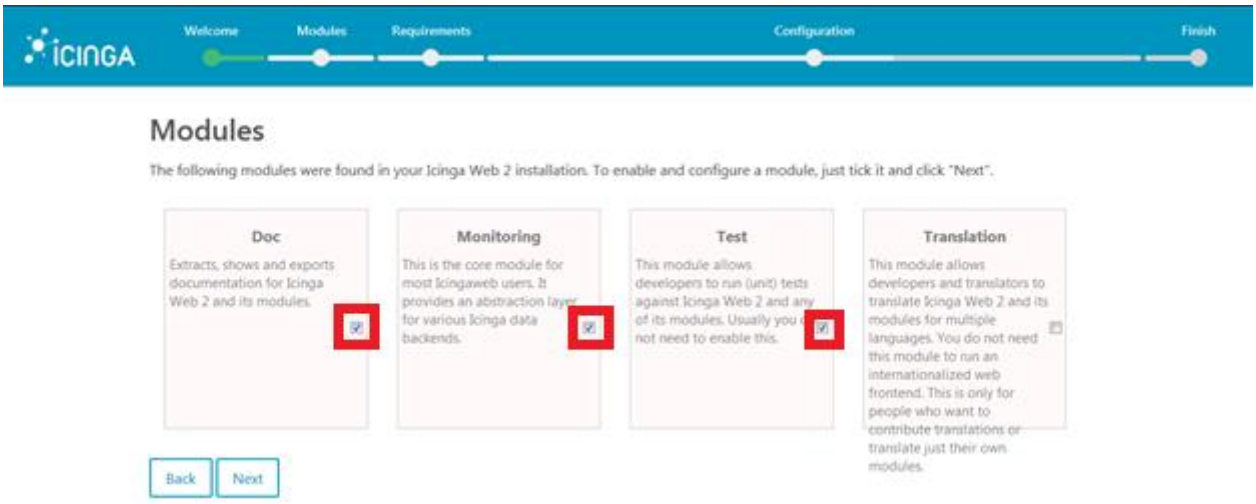
```
root@master:~#addgroup --system icingaweb2

addgroup: The group `icingaweb2' already exists as a system group.

Exiting.
```

```
root@master:~# usermod -a -G icingaweb2 www-data

root@master:~#icingacli setup config directory --group icingaweb2;
Successfully created configuration directory /etc/icingaweb2
root@master:~# icingacli setup token create;

The newly generated setup token is:2864d3be22220d56
```

Paste the generated token and click on Next button

5.1.6.2 Selecting modules

On next page, the modules to be installed are chosen. Select: Doc, Monitoring and Test



5.1.6.3 Verifying the PHP settings

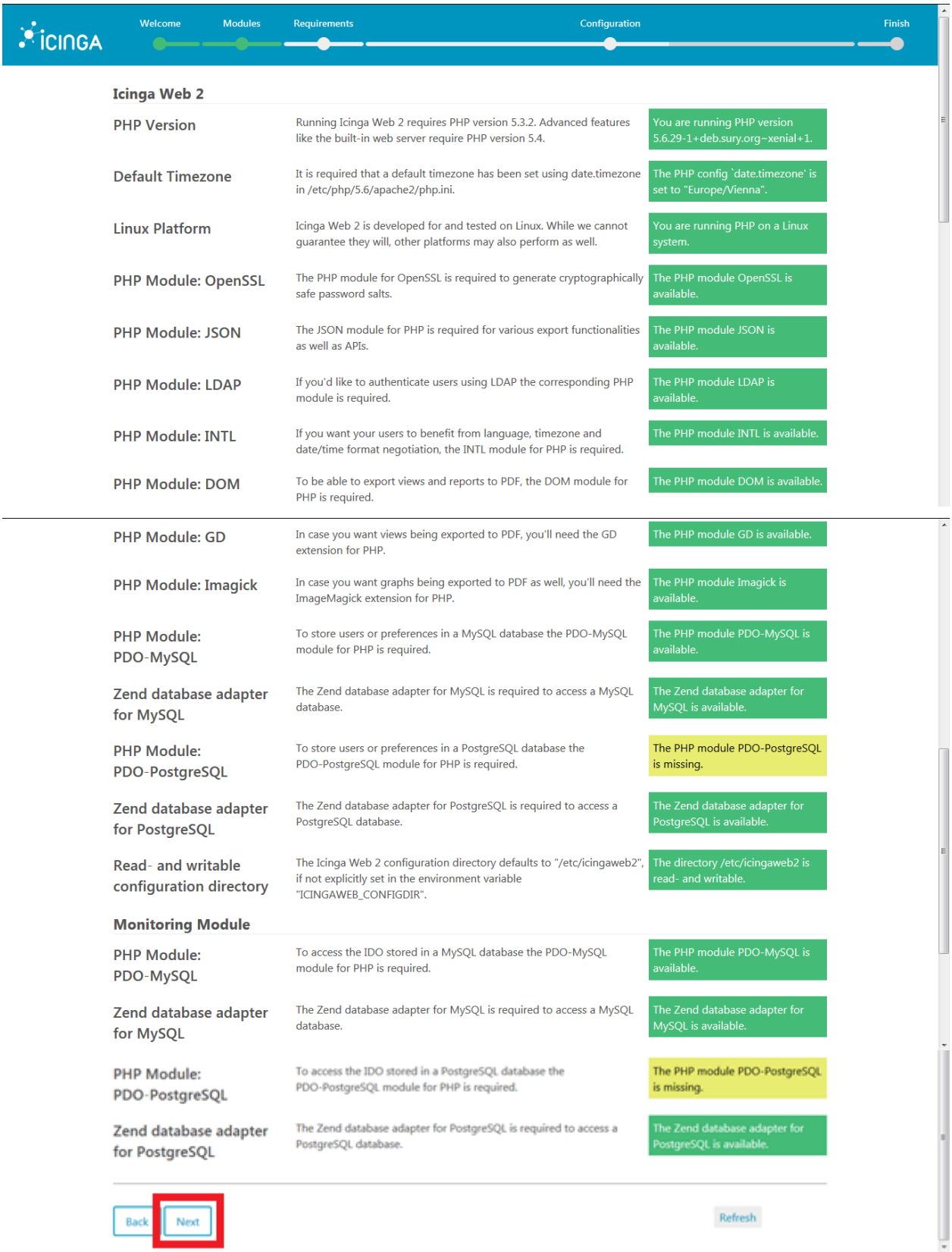
On next page, if any PHP module is missing, execute:

```
root@master:~#apt-get install php5.6-gd php5.6-json php5.6-dba
php5.6-intl php5.6-ldap php5.6-pdo-mysql php5.6-imagick php5.6-dom
```

Set the proper time zone in PHP configuration file (php.ini) allocated in /etc/php/5.6/apache2. For example:

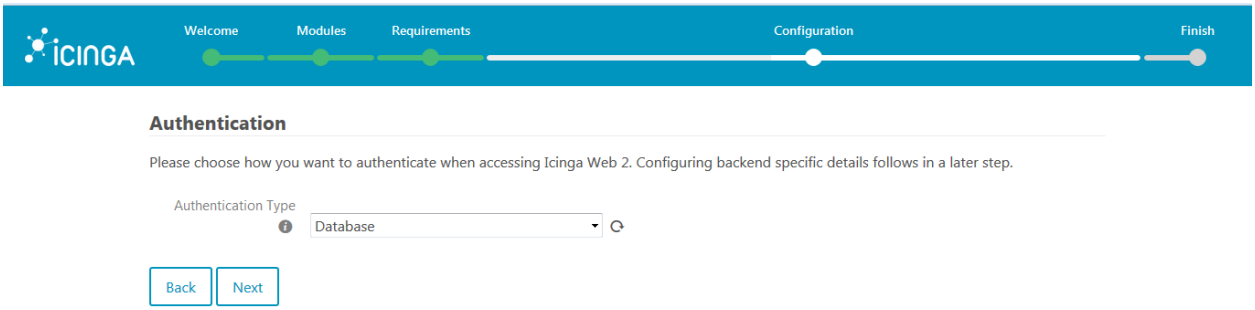
```
date.timezone = "Europe/Madrid"
```

When all issues have been fixed, click on Next button at the bottom of the page:



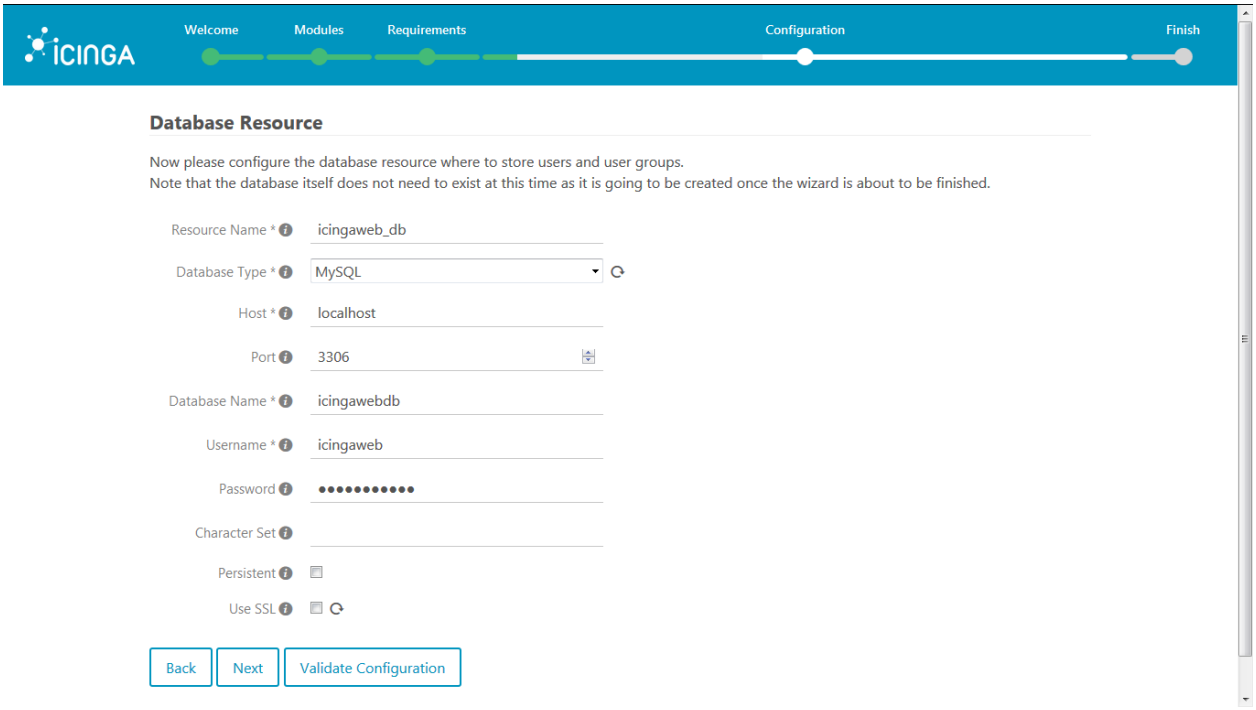
5.1.6.4 Authenticating methods

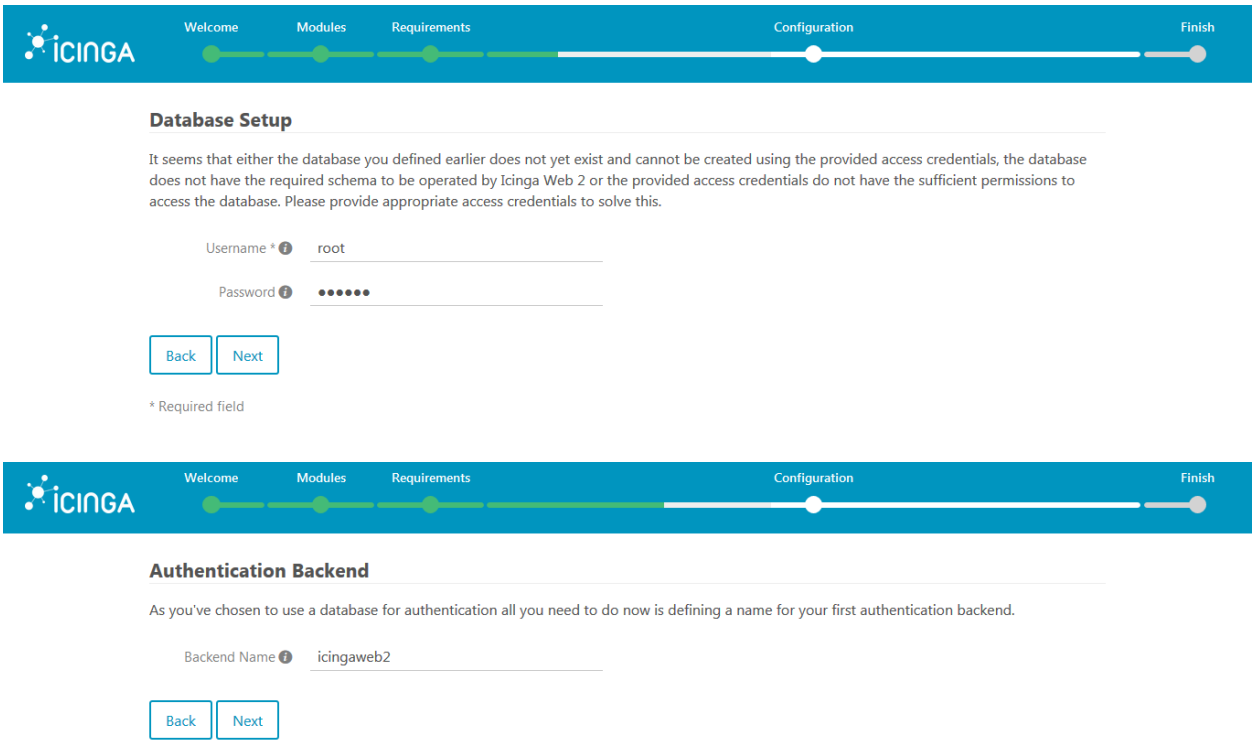
Select database authentication and click Next button



5.1.6.5 Fill out the database details that is used for authentication

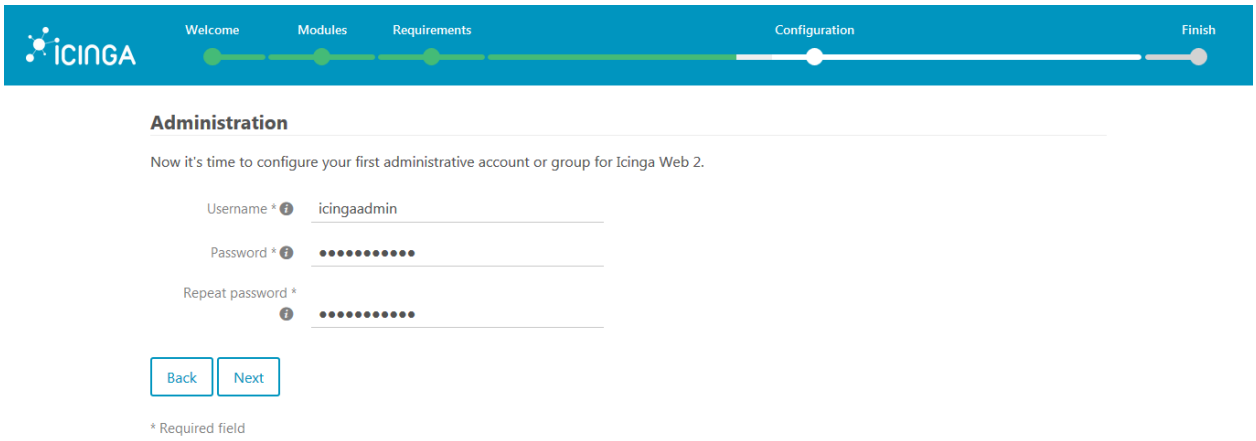
Enter the database details for the one created for web interface. Be sure that the user has enough privileges





5.1.6.6 Creating Icinga Web administration users to manage the interface

Create the Administrative account for managing the Icinga2 Web interface. Provide a username and password to manage the interface.

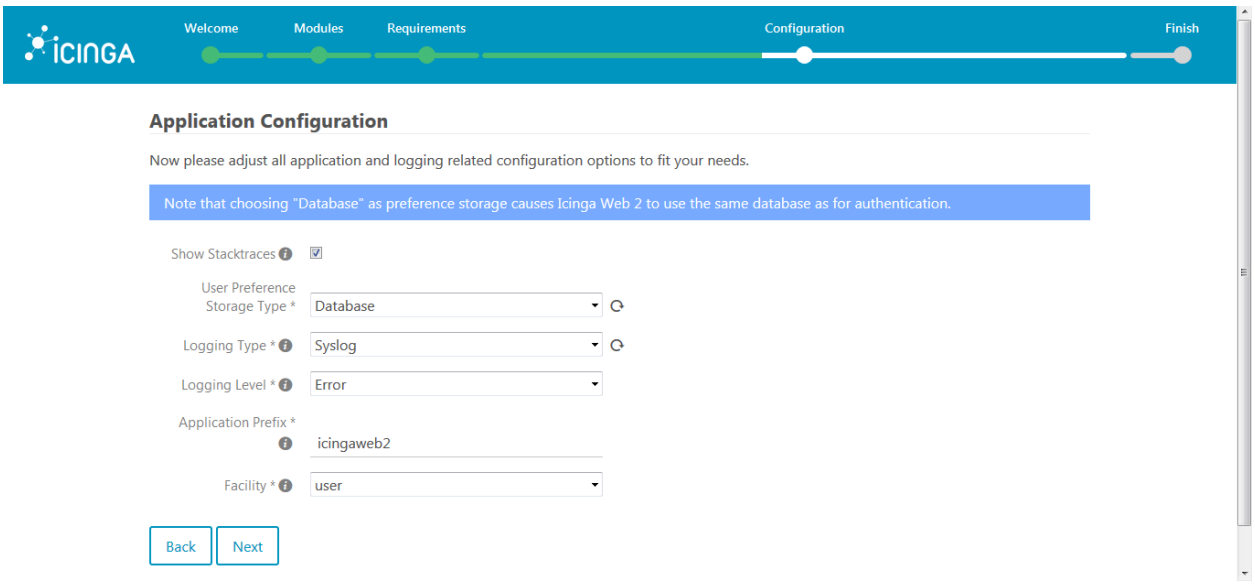


5.1.6.7 Choosing the Application configuration options.

Configure the application with these values:

- Show stacktraces: yes
- User preference storage type: Database
- Logging type: Syslog
- Logging level: Error
- Application prefix: icingaweb2

- Facility: user

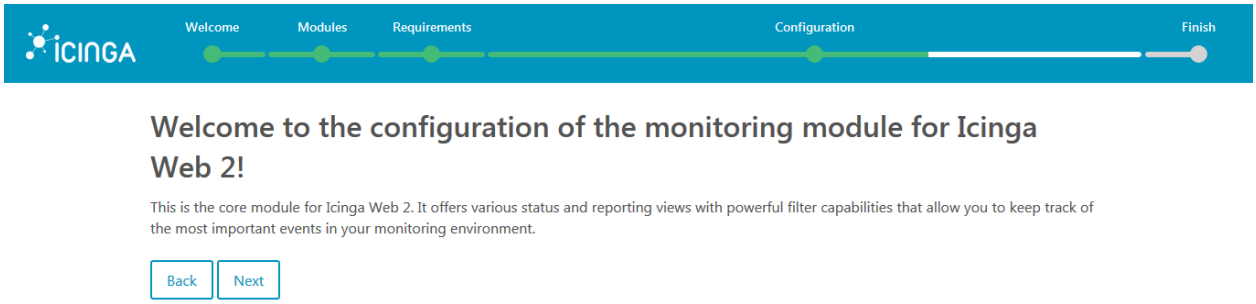


5.1.6.8 Reviewing all chosen settings

A summary of the configuration selected on previous steps is shown. If all is fine, click on Next button

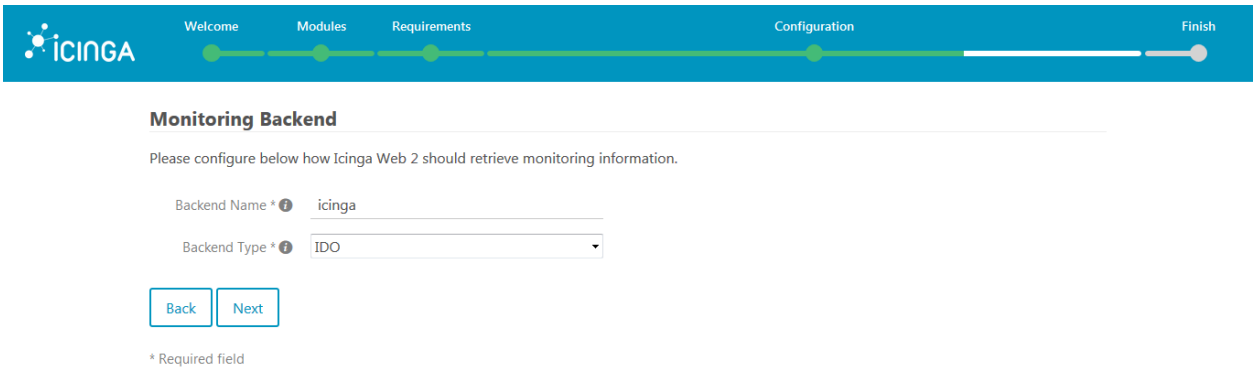
5.1.6.9 Configuring Monitoring Module.

Completed the authentication module of the configuration, next is the Monitoring module.



Select these values for Monitoring backend:

- Backend name: icinga
- Backend type: IDO

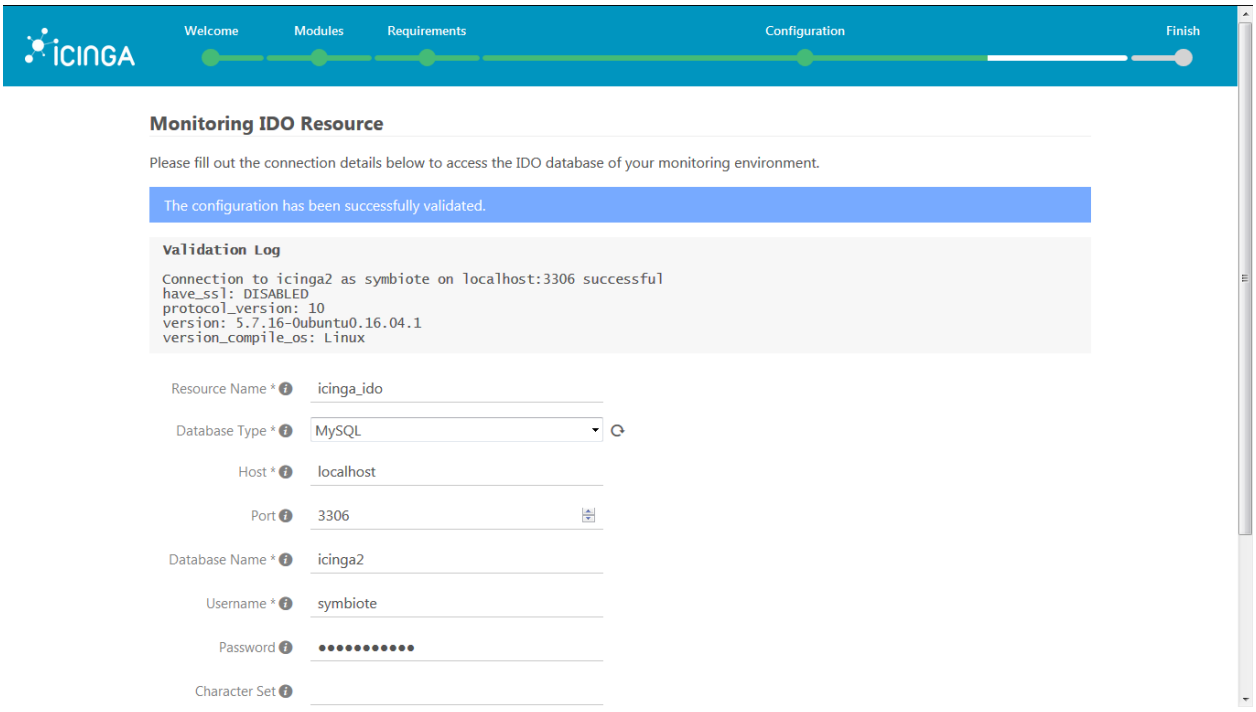


Before continue with the next screen, we have to import the icinga2 schema to the database created for Monitoring and restart icinga2 service:

```
root@master:~#mysql -u root -psecret icinga2 < /usr/share/icinga2-ido-mysql/schema/mysql.sql
```

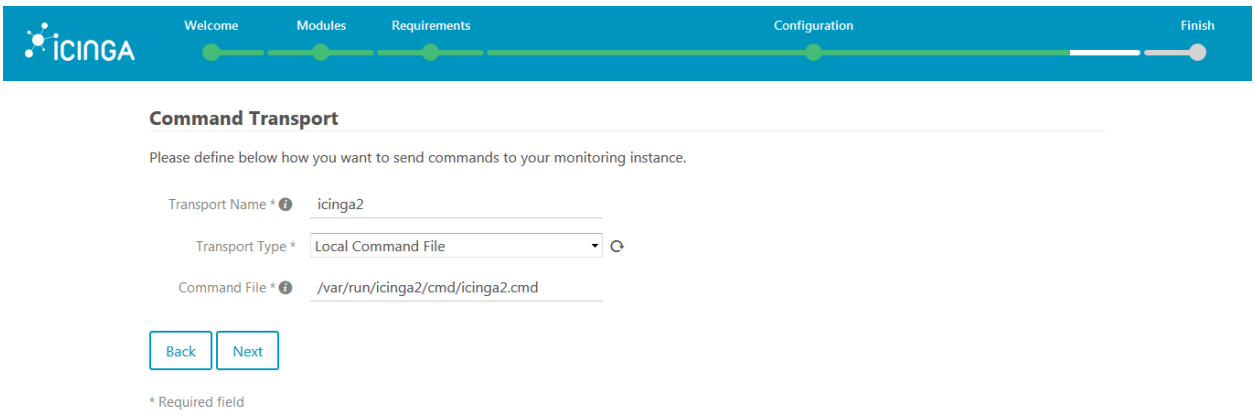
```
root@master:~#service icinga2 restart
```

On Monitoring IDO Resource page select icinga_ido for Resource name field, and complete the other fields with the information of the database created. Persistence check must be enabled.

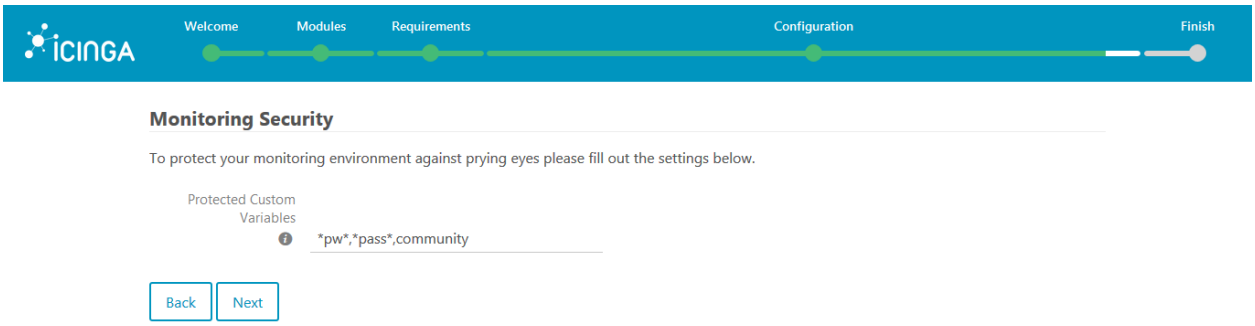


Select these values for Command Transport screen:

- Transport name: icinga2
- Transport type: Local Command File
- Command file: /var/run/icinga2/cmd/icinga2.cmd

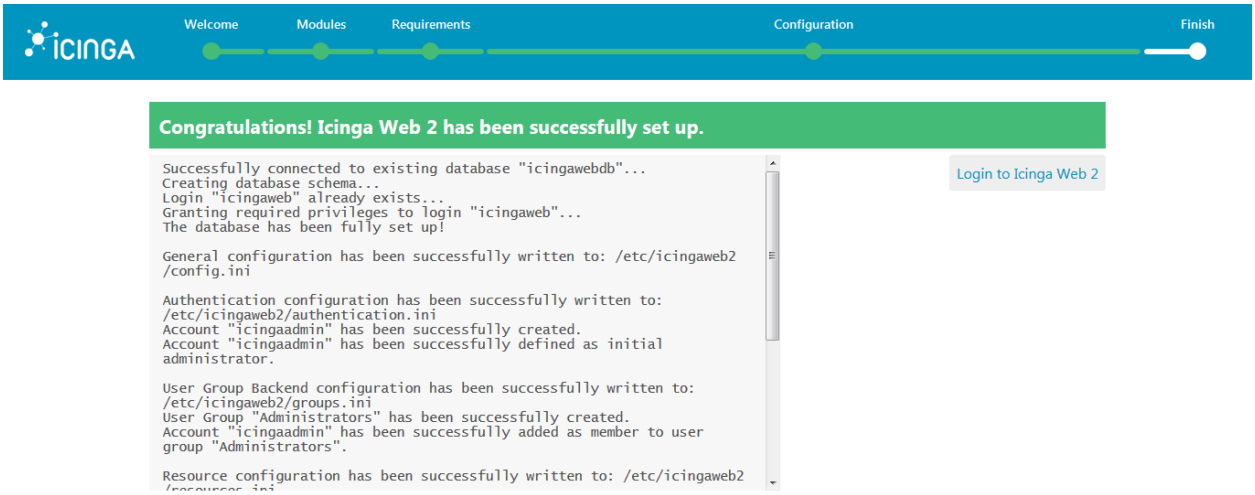


Keep the Monitoring Security screen as it appears and click Next:



5.1.6.10 Reviewing the Monitoring module configuration options

A summary of the Monitoring module configuration is shown, if all is fine, confirm it and complete the setup



5.1.6.11 Login to the Web interface

To login to the icinga2 web interface browse to

`http://<master_ip_address>/icingaweb2/`

5.1.7 Adding symbiote user to API

symbiote user must be added to icinga2 configuration in order to platform monitoring component will be able to do operations (add, delete, update devices) using the REST API provided by icinga2.

To do this task, modify the /etc/icinga2/conf.d/api-users.conf file, add following the text to the file:

```
objectApiUser "symbiote" {
password = "symbiote123"
permissions = [ "*" ]
}
```

After of this, restart icinga2 service:

```
root@master:~#service icinga2 restart
```

5.2 Adding new host on Icinga2 master to monitor

Icinga2 uses its own way of communication, using a unique and secure protocol to client and server communication by TLS with certifcates and full IPv4 and IPv6 support.

Icinga2 master, satellite and client instances use port 5665 for default TCP communication.

5.2.1 Setup the Master for remote clients

Run on Icinga2 masteras **root** user the CLI command

```
root@master:~# icinga2 node wizard
```

to setup a new node on the master. Answer the first question **n** and simply hit **Enter** until the installation finish.

```
root@VTSS031:~# icinga2 node wizard
Welcome to the Icinga 2 Setup Wizard!

We'll guide you through all required configuration details.



Please specify if this is a satellite setup ('n' installs a master setup) [Y/n]: n
Starting the Master setup routine...
Please specify the common name (CN) [VTSS031.cs1local]:
Checking for existing certificates for common name 'VTSS031.cs1local'...
Certificate '/etc/icinga2/pki/VTSS031.cs1local.crt' for CN 'VTSS031.cs1local' already existing. Skipping certificate generation.
Generating master configuration for Icinga 2.
information/cli: API user config file '/etc/icinga2/conf.d/api-users.conf' already exists, not creating config file.
'api' feature already enabled.
information/cli: Dumping config items to file '/etc/icinga2/zones.conf'.
warning/cli: Backup file '/etc/icinga2/zones.conf.orig' already exists. Skipping backup.
Please specify the API bind host/port (optional):
Bind Host []:
Bind Port []:
warning/cli: Backup file '/etc/icinga2/features-available/api.conf.orig' already exists. Skipping backup.
information/cli: Updating constants.conf.
warning/cli: Backup file '/etc/icinga2/constants.conf.orig' already exists. Skipping backup.
information/cli: Updating constants file '/etc/icinga2/constants.conf'.
information/cli: Updating constants file '/etc/icinga2/constants.conf'.
information/cli: Updating constants file '/etc/icinga2/constants.conf'.
Done.

Now restart your Icinga 2 daemon to finish the installation!
```

The setup wizard will do the following tasks:

- Verify that API feature is enabled and if not, it will generate a new local CA in the path /var/lib/icinga2/ca or use the existing one.
- Generate a new CSR, it will be signed with the local CA and copied to /etc/icinga2/pki path
- Generate a local zone and endpoint configuration for this master based on FQDN
- Set the NodeName and TicketSalt constant in constants.conf file

To verify that all has been done:

1. Check Master NodeName and TicketSalt constant executing the CLI command: `egrep 'NodeName|TicketSalt' /etc/icinga2/constants.conf`

```
root@VTSS031:~# egrep 'NodeName|TicketSalt' /etc/icinga2/constants.conf
const NodeName = "VTSS031.cs1local"
const TicketSalt = "61164a7310278d9283378ad47d496245"
```

2. Check /etc/icinga2/zones.conf file appears like below, if not please modify it (NodeName: icinga2 master server host name)

```
root@VTSS031:~# cat /etc/icinga2/zones.conf
/*
 * Generated by Icinga 2 node setup commands
 * on 2017-04-07 22:33:19 +0000
 */

object Endpoint "VTSS031.cs1local" {
}

object Zone ZoneName {
        endpoints = [ "VTSS031.cs1local" ]
}
```

The setup does not automatically restart Icinga2, so restart Icinga2 manually and verify that all is running fine:

`root@master:~# service icinga2 restart`

`root@master:~# service icinga2 status`

```
root@VTSS031:~# service icinga2 restart
root@VTSS031:~# service icinga2 status
● icinga2.service - LSB: icinga2 host/service/network monitoring and management system
   Loaded: loaded (/etc/init.d/icinga2; bad; vendor preset: enabled)
   Active: active (running) since Fri 2017-04-07 22:41:11 UTC; 4s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 13918 ExecStop=/etc/init.d/icinga2 stop (code=exited, status=0/SUCCESS)
  Process: 13927 ExecStart=/etc/init.d/icinga2 start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/icinga2.service
           ├─13908 /usr/lib/nagios/plugins/check_ping -4 -H 7.5.8.8 -c 200,15% -w 100,5%
           ├─13909 /bin/ping -n -U -w 10 -c 5 7.5.8.8
           ├─13986 /usr/lib/x86_64-linux-gnu/icinga2/sbin/icinga2 --no-stack-rlimit daemon -d -e /var/log/icinga2/icinga2.err
           └─13995 /usr/lib/x86_64-linux-gnu/icinga2/sbin/icinga2 --no-stack-rlimit daemon -d -e /var/log/icinga2/icinga2.err
```

Generate secure ticket salt for CSR Auto-Signing on Icinga2 server for Remote Client executing the CLI command:

`icinga2pki ticket --cn '<remote_client_hostname>'`

```
root@VTSS031:~# icinga2 pki ticket --cn 'zabbix.atos.net'
43baefff3a2709c03b41940328f4c2765042b216
```

The generated value is needed in the installation of the remote client, so please, keep it

5.2.2 Configuring the new host for Icinga2

Install Icinga2 from Linux distribution's package and be sure that the running system has installed openSSH package, because is needed to handle SSL communication. The installation must be done with **root** user. Then run the following CLI command in client

`root@client:~# apt-get install icinga2`

If the following message appears:

```
root@zabbix:~# apt-get install icinga2
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package icinga2
```

Is needed to add the icinga2 repository to the system: `sudo add-apt-repository ppa:formorer/icinga`

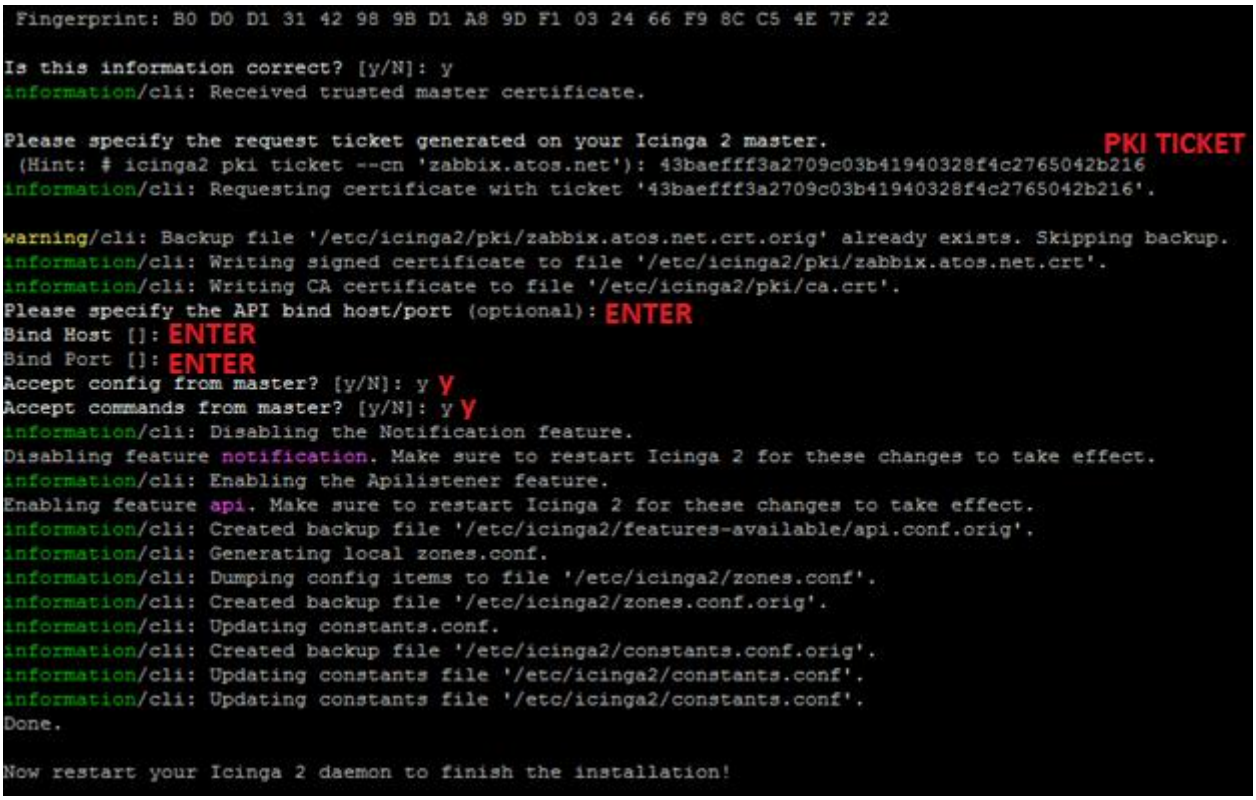
If the system does not recognize the previous command, is needed to be installed using the command`apt-get install software-properties-common`. After of this, add the icinga2 repository.

```
root@zabbix:~# add-apt-repository ppa:formorer/icinga
 This PPA provides Icinga 1, Icinga 2 and Icinga web Packages for Ubuntu. They are directly derived from the Debian Packages that I maintai
n within Debian.
 More info: https://launchpad.net/~formorer/+archive/ubuntu/icinga
Press [ENTER] to continue or ctrl-c to cancel adding it

gpg: keyring `/tmp/tmpqhv2mfq8/secring.gpg' created
gpg: keyring `/tmp/tmpqhv2mfq8/pubring.gpg' created
gpg: requesting key 36862847 from hkp server keyserver.ubuntu.com
gpg: /tmp/tmpqhv2mfq8/trustdb.gpg: trustdb created
gpg: key 36862847: public key "Launchpad PPA for Alexander Wirt" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
OK
```

Execute the `apt-get update` command to download the package lists from the repositories and then install icinga2 executing `apt-get install icinga2`

Install Icinga2 Node wizard into Remote Linux Host (in red appears information that must be filled during the process):`icinga2 node wizard`

```
root@zabbix:~# icinga2 node wizard
Welcome to the Icinga 2 Setup Wizard!

We'll guide you through all required configuration details.



Please specify if this is a satellite setup ('n' installs a master setup) [Y/n]: ENTER
Starting the Node setup routine...
Please specify the common name (CN) [zabbix.atos.net]: ENTER
Please specify the master endpoint(s) this node should connect to: ENTER
Master Common Name (CN from your master setup): VTSS031.cs1local MASTER HOSTNAME
Do you want to establish a connection to the master from this node? [Y/n]: Y
Please fill out the master connection information: ENTER
Master endpoint host (Your master's IP address or FQDN): 37.48.90.65 MASTER IP
Master endpoint port [5665]: ENTER
Add more master endpoints? [y/N]: ENTER
Please specify the master connection for CSR auto-signing (defaults to master endpoint host):
Host [37.48.90.65]: ENTER                                                    ENTER
Port [5665]: ENTER
information/cli: Created backup file '/etc/icinga2/pki/zabbix.atos.net.key.orig'.
information/cli: Created backup file '/etc/icinga2/pki/zabbix.atos.net.crt.orig'.
information/base: Writing private key to '/etc/icinga2/pki/zabbix.atos.net.key'.
information/base: Writing X509 certificate to '/etc/icinga2/pki/zabbix.atos.net.crt'.
information/cli: Fetching public certificate from master (37.48.90.65, 5665):

Certificate information:

 Subject:     CN = VTSS031.cs1local
 Issuer:      CN = Icinga CA
 Valid From:  Jan 13 10:00:29 2017 GMT
 Valid Until: Jan 10 10:00:29 2032 GMT
```



The setup wizard will do the following tasks:

- Generate a new self-signed certificate and copy it into the /etc/icinga2/pki path
- Store the master's certificate as trusted for requesting a new signed certificate
- Request a new signed certificate from the master and store the certificate and the master CA in /etc/icinga2/pki
- Generate a local zone and endpoint configuration for this client and the provided master information
- Disable the notification feature for this client
- Enabling the API feature and setting optional bind_host and bind_port
- Set the NodeName constant in constants.conf

Verify that the configuration has been modified successfully:

- NodeName and ZoneName are correct executing

  `grep '<remote_client_hostname>' /etc/icinga2/constants.conf`

  ```
  root@zabbix:~# grep 'zabbix.atos.net' /etc/icinga2/constants.conf
  const NodeName = "zabbix.atos.net"
  const ZoneName = "zabbix.atos.net"
  ```

- Zones.conf file should be appears like this:

```
root@zabbix:~# more /etc/icinga2/zones.conf
/*
 * Generated by Icinga 2 node setup commands
 * on 2017-04-08 23:04:54 +0200
 */

object Endpoint "VTSS031.cs1local" {
        host = "37.48.90.65"
        port = "5665"
}

object Zone "master" {
        endpoints = [ "VTSS031.cs1local" ]
}

object Endpoint NodeName {
}

object Zone ZoneName {
        endpoints = [ NodeName ]
        parent = "master"
}
```

The wizard does not restart icinga2 automatically, so restart it manually and check that all is working fine

```
root@client:~# service icinga2 restart
root@client:~# service icinga2 status
```

5.2.3 Discovering client services on the Master

Icinga 2 clients sync their locally defined objects to the master node. Nodes can be listed, added, filtered and removeed based on their node, zone, host or service name.

Execute `icinga2 node list` to view the nodes registered on master node

```
root@VTSS031:~# icinga2 node list
Warning: CLI command 'node list' is DEPRECATED! Please read the Changelog.

Node 'zabbix.atos.net' (last seen: Sat Apr  8 21:18:01 2017)
    * Host 'zabbix.atos.net'
        * Service 'apt'
        * Service 'disk'
        * Service 'disk /'
        * Service 'http'
        * Service 'icinga'
        * Service 'load'
        * Service 'ping4'
        * Service 'ping6'
        * Service 'procs'
        * Service 'ssh'
        * Service 'swap'
        * Service 'users'
```

This information is not updated automatically when a new remote client is added, so is needed to run the `icinga2 node update-config` in order to update all the client services on master so the required configurations are generated.

The following tasks are performed:

- Add Endpoint and Zone objects for the new added host
- Add cluster-zone health check for the master for reachability and dependencies
- Use the default templates satellite-host and satellite-service defined in /etc/icinga2/conf.d/satellite.conf
- Apply a dependency for all other hosts on the remote satellite prevening failure checks/notifications

After updating the configuration repository, make sure to reload icinga2 executing the CLI command: `service icinga2 reload`

Then go to Icingaweb2 and the new added remote host appears:

6 Acronyms

API	Application Programming Interface
ATOS	ATOS Spain SA
DB	Database
CLI	Command line interface
CSR	Certificate Signing Request
EC	European Commission
EU	European Union
FQDN	Fully qualified domain name
HTTP	Hypertext Transfer Protocol
ICOM	Intracom Sa Telecom Solutions
IOSB	Fraunhofer Gesellschaft zur Förderung der Angewandten Forschung ev
IoT	Internet of Things
IP	Internet protocol
ISP	Internet Service Provider
KIT	Karlsruhe Institute of Technology
MQ	Message queue
NAVIGO	Na.Vi.Go. SocietaConsortile a ResponsabilitaLimitata
NWX	Nextworks
OSS	Open Source Software
PHP	PHP: Hypertext Preprocessor
R&D	Research and Development
REST	REpresentational State Transfer
S&C	Sensing & Control Systems SL
SQL	Structured Query Language
SSL	Secure sockets layer
symbIoTe	Symbiosis of Smart Objects across IoT Environments
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UC	Use Case
UNIVIE	Universität Wien
TCP	Transmission Control Protocol

WP	Work Package