Icinga and Oracle Part3 - Configuration

This how to focuses on configuration issues

Part1: Icinga and Oracle Part1 - Installing Oracle Software

Part2: Icinga and Oracle Part2 - Building software

Part3: Icinga and Oracle Part3 - Configuration

Part4: Icinga and Oracle Part4 - Monitoring Oracle

Part3 Configuration Steps

configure icinga base without oracle and idoutils

A good practice is to get base Icinga configuration running without any specials. Follow the instructions in Icinga documentation http://docs.icinga.org/latest/en/ch03.html. Only a small number of hosts and services is required. This should proof icinga base is running well and we need some data sources for our database.

provide Oracle server with icinga user and schema objects

Although it is possible to have a flat installation which is included in the distribution from a DBA point of view you should consider additional thoughts on your Oracle design. You can check with this thread:http://www.nagios-portal.org/wbb/index.php?page=Thread&threadID=21860. I will follow these rules.

Oracle Environment checks

- Your database is up and running
- You have set Oracle variables in your environment. It depends of your client type (see FullClient Environment or InstantClient Environment)
- You are able to connect to your database with Oracle sqlplus utility as sys user (ee Oracle Connectivity Tests)

If not, please correct errors first. ***it makes no sense to proceed without this.***

- you need access to an administrative Oracle user (DBA-role). If you want to give grants on v$views for monitoring, this must be done as SYS-user only
- check your filesystem where your Icinga tablespace files should be placed. Make sure that there is enough space and oracle:dba owns this. Notice the exact locations
- check the values you provided for oracle user and password given during Icinga configure (see Part2), this user you have to create

- change into oracle scripts directory <icinga_source>/module/idoutils/db/oracle

create tablespaces and user

- as of icinga1.4 the scripts are prepared to distribute data, index and lobs into different tablespace.Also an sample script "create_oracle_sys.sql" for creating user and tablespaces is supplied into distribution. You can fill your database settings into a parameter file **icinga_defines.sql**
  Edit the script for your needs. It defines variables for filesystem and user. You have to **change** these values! if you don't want a distributed file system layout, you can set all tablespaces to the same name and remove obsolete tablespace creation sql from **create_oracle_sys.sql** script
- **create_oracle_sys.sql**will do
  - create a spool (log) file which will record all steps
  - do error handling. The script will exit on the first error
  - create a data and index tablespace with a small (default) extent size, autoextensible depending on your variables
  - create the icinga ido2db user "icinga"
  - grant quotas on the newly created tablespaces
  - grant needed system rights

create session
create table
create procedure
create trigger
create sequence
create synonym
create view
create type
alter session

  - grant recommanded monitoring views. must be grantet direct from SYS

select on v_$session
select on v_$process
select on v_$sesstat
select on v_$mystat
select on v_$statname

- run the script with SYS-User (assuming script is in your current directory)

```
sqlplus "sys@<instance> as sysdba" @ create_oracle_sys.sql
```

- check the logfiles for errors. If the script terminates with errors you should check the reason and comment out statements which were already successfully executed or drop these objects before rerun. Usually if you don't do so it will stop on the first duplicate statement again.

check oracle startup parameters

usually in small environments oracle default init.ora settings are sufficient. IDO2DB will use up to 300 open cursors and 2 simultanous connections. icinga-web up to 1.4.1 will use one session per connected web user

create schema objects

After the users exists and owns the needed rights you must run the supplied object creation script (/oracle.sql) .

- check if entries in icinga_defines.sql (if exists, otherwise in oracle.sql) fit your needs
- fill in your real tablespace settings
- Attention: **it will drop all existing objects for this user!** comment out the affected lines if you don't want this
- run it in sqlplus as icinga db user within the directory contains oracle.sql

```
sqlplus <icingadbuser>@<instance> @ oracle.sql
```

Check for errors in the created log file. Correct errors, then you can run the script again. It will drop all objects and recreate them from script.

alternativ: use contributed all-in-one shell script

for your convience as of v1.4 you can find a script "create_oracledb.sh" doing both steps in one in ../scripts/. it will take use of the sys and the objects script. You need to create your own **icinga_defines.sql** or the script will create its own using hardcoded defaults.

configure idomod and ido2db

After you created the Icinga database user and schema objects you need to advice Icinga to fill in the data. This is a two step process. First, the idomod neb module will receive configured events and write them to a socket (local unix or remote tcp) . Second, a special additional service named ido2db is listening on this socket and writes the incoming data after translating into SQL to the database. For Oracle databases prepared statements are used instead of simplly building a new sqlstring for each and every data. This will speedup your database and reduce resource consumption for parsing and saving space in memory.
If you don't have already done, install idoutils out of your source tree. This will install idomod and config to the predefined locations. Most of the settings can be defined within Icinga configure options

```
make install-idoutils
```

register idomod in icinga.cfg

This will tell Icinga to enable event broker and use idomod to write out these events. You will find this in section "EVENT BROKER MODULE" and it needs to be enabled and configured with the correct paths for module and config.

```
event_broker_options=-1
broker_module=/opt/icinga/bin/idomod.o config_file=/opt/icinga/etc/idomod.cfg
```

change idomod.cfg to printout data

If you intend to run idoutils not on the same host on which Icinga itself runs, you have to use tcp connections. On the same machine I suggest to use unix socket connection method. You need to record this, because you need exact the same settings for ido2db

```
output_type=unixsocket
output=/opt/icinga/var/ido.sock
```

Additionally you need to tell idomod what it should report and define debugging settings. In the beginning I recommend to set debugging level to the highest possible. This requires to allow a big debug file (default 1M is too small). If all is working well, you might reduce the level and the file size.

```
#data options
data_processing_options=67108861 #all without timed statistics
config_output_options=2 #dump config data (host/services/checks...) when restarting
#debug setting
debug_level=-1 #all
debug_verbosity=1 #medium
debug_file=/opt/icinga/var/idomod.debug #to look into
max_debug_file_size=50000000 #50M
```

change ido2db.cfg for oracle backend

Based on the settings in idomod.cfg we have to change ido2db.cfg. I will use my settings from above. We should set debugging option to higher values as well. From the Oracle point of view we have to provide database connection settings for the Icinga schema we created. Host and port will be ignored, use correct database connect string in dbname.

```
#connection
socket_type=unix
socket_name=/opt/icinga/var/ido.sock
...
#oracle specific
db_servertype=oracle
db_name=<connection string>
db_user=<icingadbuser>
db_pass=<icingadbpass>
...
#debug settings
debug_level=-1 #anything
debug_file=/opt/icinga/var/ido2db.debug
max_debug_file_size=50000000
```

add oracle environment to ido2db start script

You can easily add Oracle environment directly into the ido2db init script or on redhat based systems to the sourced ido2db sysconfig file (/etc/sysconfig/ido2db). This should be the same we used in Part2 while creating ocilib. See FullClient Environment or InstantClient Environment. Please note: we need american locale settings (NLS_LANG)

```
vi /etc/sysconfig/ido2db
# User specific environment and startup programs
ORACLE_HOME=/opt/oracle/instantclient_11_2
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME
export PATH=$PATH:$ORACLE_HOME
export TNS_ADMIN=/var/opt/oracle #only needed if you cant use oracle default network settings
export NLS_LANG=american_america.utf8
```

Now we have to register the services within our system startup if it should be activated this way. Do not start it now. The following is only true with RedHat-based linux.

```
chkconfig --add ido2db
chkconfig --add icinga
chkconfig on ido2db
chkconfig on icinga

#check it
chkconfig --list ido2db
ido2db          0:off   1:off   2:on    3:on    4:on    5:on    6:off
 chkconfig --list icinga
icinga          0:off   1:off   2:off   3:on    4:on    5:on    6:off
```

activate it

To activate change settings we should stop all Icinga processes first. Then **first** start **ido2db** then Icinga and take a log at the debug files

```
service ido2db start # or /etc/init.d/ido2db start or whatever fits on your system
service icinga start # or /etc/init.d/icinga start or whatever fits on your system
```

check it

You should see activities in the defined .debugfiles, but no errors. The following will grep for oracle error messages

```
grep ORA /opt/icinga/var/ido2db.debug #modify path with your location
```

For connection errors see Part1

Next check if your configured host and services are provided from Icinga.
This we will do with sqlplus. This should return the same count you have in icinga shown by the classic gui. If count=0 and if your config is defined as shown and no errors found in logfiles you may have started Icinga before ido2db. Be aware that starting ido2db will clear configuration information first and reloaded after starting icinga with idomod.

```
sqlplus <icingadbuser>/<icingadbpw>@<instance>
select count(*) from hosts;
COUNT(*)
----------
        15
select count(*) from services;
 COUNT(*)
----------
        37
```

After executing some checks you should find entries in the database as well

```
--sample count
select count(*) from servicechecks;
 COUNT(*)
----------
      4234
```

Now you are done and can proceed with Icinga web optionally which is based on these database entries in contrast to the classic gui which is file based only.

configure icinga web addon for oracle

Icinga web is the new php based web frontend for icinga. This means, you will need a recent (5.2.5+) php stack and an Oracle connector. At this time, this is the oci8 extension. See Part 2 of this howto for hints installing oci8 and building icinga web. You should make sure that your php and oci checks were successfully. **It makes no sense to continue without this!**

create icinga web database user

Icinga web needs a separate schema other than the Icinga ones.
I highly recommend to create a separate user for this.
Attached is the script create_icingaweb_sys.sql which will do the job. Edit this to suit your needs first. It will use the tablespaces we defined for Icinga as well.

- run the script with SYS-User (assuming script is in your current directory)

```
sqlplus "sys@<instance> as sysdba" @ create_icingaweb_sys.sql
```

create icinga web schema objects

Now we need schema objects as well. You can implement this manually and use "make db-initialize". Original SQL-Scripts for each database you can find in <icingaweblocation>/etc/schema/.
But I am prefering a more specific layout and object handling as scripted in "create_icingaweb_objects.sql" as of icinga-web V1.3 posted in http://www.monitoring-portal.org/wbb/index.php?page=Thread&postID=146953#post146953 and attached above for V1.7.0 You should check for update scripts in etc/schema/updates when using higher versions afterwards.

- save the script to your current directory
- fill in your real tablespace settings again.
- Attention: **it will drop all existing objects for this user!** comment out the affected lines if you don't want this
- run it in sqlplus as icingaweb db user.

```
sqlplus <icingawebuser>@instance @ oracle.sql
```

start it

Configuration is already done when you provided all configure options. If you didn't specify all these options, you will need to change affected xml files directly. If ready, reload your web server to bring your http changes into effect. Now you should be able to point your browser to http://<yourhost>/icinga-web. Initial credentials are "root" with password "password".
If it won't work, first take a look into <icingaweblocation>/app/config/database.xml if you get errors while starting icinga web.

additional hints:

- configure options named "with-db-*" means icinga web user db account. Type should be "oracle".
- configure options named "with-api-db*" means icinga idoutils account. Type should be oci8" up to V1.4, later "icingaOracle".
- configure option --with-api-db-prefix must set to empty string
- apache environment must know about oracle environment, at least LD_LIBRARY_PATH and TNS_ADMIN. Check again if oci8 extension works with supplied credentials (see oci8 build and check)
- if you get an ORA-1861 Error you should add this to database.xml maybe below charset parameter line (https://dev.icinga.org/issues/1284).This issue has been fixed with V1.6

```
<ae:parameter name="date_format"><![CDATA[YYYY-MM-DD HH24:MI:SS]]></ae:parameter>
```

To view more information about API-Problems i removed temporary the appkit defined exception handler in <icingaweblocation>/app/modules/AppKit/lib/logging/AppKitExceptionHandler.class.php (three commented lines)

```
public static function initializeHandler() {
#               self::$oldExceptionHandler =
set_exception_handler(self::$handlerException);
#               self::$oldErrorHandler = set_error_handler(self::$handlerError);
#               ini_set('display_errors', false);

        }
```

After changing any of these xml files you need to clear the cache first before retrying. See icinga web documentation for details.

```
<icingaweblocation>/bin/clearcache.sh
```

You can find useull error messages for icingaweb in <icingaweblocation>/app/data/log and the icinga api logfile in <icinga>/share/icinga-api/log/icingaAPI.log

Continued in Part4: Icinga and Oracle Part4 - Monitoring Oracle Instances