docker-zabbix-script-sender Documentation

Release latest

Moreau Cyril

August 24, 2016

Contents

Contents:

CHAPTER 1

Docker Zabbix Script Sender (DZSS)

1.1 Description

The DZSS docker container is a zabbix agent that send zabbix trap to a zabbix server. It is a script based monitoring, that allows you to develop your own scripts to monitor whatever you want in docker containers. The python docker API is already provided to be able to get any information about docker. With this container you will be able to send zabbix traps thanks to scripts developped in the language you like (bash,python,perl).

1.2 Overview

This Tool contains 3 components :

- Docker Monitoring Manager (DMM)
- The Trap Sender Component (TPC)
- The Configuration Parser Component (CPC)

Each components has a specific usage.

- **Docker Monitoring Manager (DMM) :**

This component is a thread that take care of every docker containers in the host. Every "time interval" in second (By default every 30min), this component check if a running container is in the configuration file and if the component has to monitor it.

At the start, the Component identify every containers on the host. Then check in the Default.ini Configuration file if there is a section related to the container (if container's name is a section in the file)

If it does exist, a new thread "Trap Sender" is created to send statistic to the zabbix server. Otherwise, it checks with the next docker container.

This Component manage the Trap Sender component. Every time there is a new statistic to send to the server, The DMM create a TPC thread according to the settings in the configuration file (provided by CPC Component).

Then TPC will get the information and send it to the server. Also, It is checking periodically if the configuration file has been change and change the TPC thread settings if needed. And finally, DMM shutdown the TPC thread when you don't need it anymore (if the docker container is shutdown or if the statitic in the configuration file has been deleted).

In the log file, everything related to this component will get the Logger Name "MAIN"

- **Trap Sender Component (TPC) :**

This component is a thread that take care of the Trap that will be send to zabbix server. As I said, the TPC thread is created, setup and launch by the DMM Component. And can be periodically modify if you change the settings in the configuration file, the TPC Component will change its settings on the fly. The role of the TPC Component is to get the information from the different docker container´s and send the results to the ZABBIX SERVER to monitor them.

One TPC Thread is created for each stat we want to get. One stat is corresponding to one script. When the DMM create the TPC, it gets from the configuration file parameters that will allow us to get the stat we need about a docker container.

The parameter of the TPC Thread are :

- **script/command** to get the information from the monitored docker container
- **delay** corresponding to the delay that the process is waiting periodically before to execute it again
- **Key** corresponding to the key setup on the ZABBIX SERVER in the ITEM settings with a AGENT ZABBIX (ACTIVE) type of trap

In the log file, everything related to this component will get the Logger Name "TRAPSENDER" or "COMMAND"

- **Configuration Parser Component (CPC) :**

This Component manages the Configuration File.

By Default there is only one Default.ini configuration file, where you can setup everything about the configuration. But for more flexibility the Default.ini file can reference other files and you can order your scripts, config file and docker container as you want. You can create a complex configuration file hierarchy with nested config file. More explanation in the section dedicated to the CPC.

This Component parse every config files and send the information to the DMM, that will use it to monitor CPC parse only JSON files.

These files contain different sections, subsections and attributs :

- **CONFIG** (section) link a DZSS Container with a configuration file. Indeed you can have one file per monitoring DZSS Container
- **Container's name** (section) contains information as reference and script subsection. the section name has to be the same as the container name that you want to monitor.
- **reference** (subSection) contains file references to be able to get more script to monitor, permit to sort the config files by type of scripts (see the CPC dedicated section)
- **script** (subSection) contains the script that will be executed by the TPC thread to get the stat and send it to ZABBIX SERVER

In the log file, everything related to this component will get the Logger Name "CONFPARSER"

The Log File logs every action, such as thread creation/change/shutdown, error of config file, traps sent, result of the trap... Moreover you have the thread id that allow you to recognize quickly what TPC thread has an issue.

1.3 As a Docker container itself

The monitoring tool is running inside a Docker container. An [automated build](https://hub.docker.com/r/troptop/docker-zabbix-script-sender/) always provides the latest stable version :

```
docker pull troptop/docker-zabbix-script-sender
```

1.4 Source File & Contribution

Source Code : https://github.com/troptop/docker-zabbix-script-sender

Issue Tracker : https://github.com/troptop/docker-zabbix-script-sender/issues

CHAPTER 2

DZSS Settings

2.1 Settings the Working Directory

The Working directory into the DZSS container is **/usr/src/app**.

So if you want to interact directly with the DZSS container you have to share this directory with a local directory on the docker host.

2.1.1 The minimum Working Directory Settings

The docker container needs at least a settings JSON file called "Default.ini" and a log directory called "logs".

```
mkdir WorkDir && cd WorkDir
mkdir logs
touch Default.ini
```

You can make a try to see if you get any errors to create the DZSS container :

```
docker run --name=zabbix-client -it -v /PATH/TO/VOLUME:/usr/src/app -v /var/run/docker.sock:/var/run/
```

When starting the container the log file **docker-zabbix-sender.log** will be created in the logs directory.

A rotation setting is setup on the log. The container create a log file every day during 7 days.

2.2 Docker Command Line Settings

The DZSS command line need only a few parameter :

- **Environment variable :**
  - **ZABBIX_SERVER** : the Zabbix Server IP or HOSTNAME
  - **INTERVAL** : The Time Interval for DMM component to check if the configuration has been change
- **Sharing Volume :**
  - **docker.sock** : the docker.sock file need to be shared to be able to communicate with docker with the docker API
  - **Working Directory** : you have to share the working directory (/usr/src/app)if you want to Configure the container

Example :

```
docker run -e ZABBIX_SERVER=IP_SERVER/HOSTNAME -e INTERVAL=30 --name=zabbix-client -it -v /PATH/TO/VO
```

2.3 File Configuration

All the configuration files are **JSON files**.

The Default configuration file is called **Default.ini**.

2.3.1 Example

The File can look like :

```
{
        "CONFIG": {
                "source":[
                        {"server": "zabbix-client1", "file":"source_zabbix1.json"},
                        {"server": "zabbix-client2", "file":"source_zabbix2.json"}
                ]
        },
        "mypostgres": {
                "reference": [
                        {"file" : "storage/database.json","section" : "postgresql"}
                ],
                "scripts" : [
                        {"key" : "postgreskey", "interpreter" : "bash", "file" : "mypostgres_check.sh
                ]
        },
        "myapache": {
                "reference": [
                        {"file" : "apache.json","section" : ""},
                        {"file" : "apache.json"}
                ],
                "scripts" : [
                        {"key" : "check_myapache","interpreter" : "bash","file" : "check_myapache.sh"
                ]
        }
}
```

2.3.2 Sections Explanation

- **CONFIG Section :** This section allow you to setup another specific file for a specific DZSS Container.

  This section contains a source subSection. This subSection contains a list of parameter corresponding to **the DZSS Container's name** and **the specific configuration file** apply to it.

  For exemple, you have 2 DZSS containers running on the same host (zabbix-client1 and zabbix-client2). The container called zabbix-client1 will use the configration file source_zabbix1.json and the container called zabbix-client2 will use the configuration file source_zabbix2.json

  The CONFIG Section can only be declared in the Default.ini file

- **Container's Name Section :** This Section allows you to setup specific scripts for the container you want to monitor. The name of the section has to have the exact same name of the container. This section has 2 subSections :

1. **reference** : This subSection references other files you want to use to monitor this container in more of the scripts you already have in this section. Useful if you want to have a specific file for a certain type of servers (exemple : a generic configuration file for all the apache servers)

   A reference has 2 parameters corresponding to the **"Configuration File"** you want to use to monitor the server and **"The Section"** in this file you have to use to monitor it

   In the example above, the subSection mypostgres contains one reference. In more of using the scripts in the scripts subSection to monitor the mypostgres container, DZSS will get the scripts in the "/usr/src/app/storage/database.json" configuration file at the "postgresql" section

   In the same way, in the subSection myapache, it has 2 references. This section will monitor the container called myapache. In this case these two references are identicals. The parameter "section" is empty or does not exist. This means that DZSS will get every section in the "/usr/src/app/apache.json" configuration file to monitor the server

With the reference files, we can imagine that the apache.json file looks like :

```
{
        "apache2.0": {
                        ....
        },
        "apache2.2": {
                        ....
        },
        "apache2.5": {
                        ....
        }
}
```

The database.json file looks like :

```
{
        "mysql": {
                        ....
        },
        "postgresql": {
                        ....
        },
        "sqlite": {
                        ....
        }
}
```

The reference section allows you to be flexible in your monitoring configuration.

2. **scripts** : This subSection list the scripts you will execute to monitor the containers. DZSS will execute this script and request the zabbix server send it the statistic.

   A script has 5 parameters corresponding to the **"key"** you had setup on the zabbix server, **"interpreter"** is the interpreter used to execute the script, **"file"** is the script file, **"argument"** are the argument passed to the script, **"delay"** is the time interval where this script will be executed periodically.

   If you do not setup up the delay or if it is empty, **the default value is 30s**

   The output format of the script has to be only one value get by an echo ouput on stdout

   Then DZSS format the result to be able to send the request to the zabbix server

The format :

```
[{'hostname' : CONTAINER_NAME, 'key' : KEY_IN_CONFIG_FILE, 'timestamp' : str(int(time.time())), 'valu
```

For example the script below for the container "mypostgresql" is a bash script called "mypostgres_check.sh" and will be called every 60s:

```
"scripts" : [
                {"key" : "postgreskey", "interpreter" : "bash", "file" : "mypostgres_check.sh", "argu
        ]
```

The script "mypostgres_check.sh" contains :

```
#!/bin/bash

echo -n 2
```

So DZSS will format the result such as :

```
[{'hostname' : 'mypostgresql', 'key' : 'postgreskey', 'timestamp' : '1451370349', 'value' : '2'}]
```

2.3.3 Specific rules

- The configuration files are **case sensitive**
- Be careful with the keys you are using. if 1 monitored container has 2 scripts with the same key, just one of these script will be used
- **Nested references** configuration files, be aware of a possible infinite loop with the references configuration file, it would crash the DZSS
- Error of json files parsing. does not have a big impact, only shutdown the TPC thread contained in the json configuration file failing, when you will correct the error in the json config file, the TPC thread will start again.