ZABBIX Administration

APPLICATION MONITORING

PostGreSQL

Follow up

|  |  |  |
| --- | --- | --- |
| Actors | Name | Functions |
| Written by | Arnauld FRACHE | Zabbix Admin |
| Checked by |  |  |
|  |  |
| Checked by |  |  |
|  |  |

Distribution list

This document has been distributed to:

|  |  |
| --- | --- |
| Name | Functions |
| On G drive | G:\EMT\03 - Projets\01 - Zabbix\01 - Documentation\01 - Doc SG |
|  |  |
|  |  |
|  |  |

Release notes

|  |  |  |
| --- | --- | --- |
| Version number | Date | Modifications |
| 1.0 | Dec 11, 2018 | Create |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
| Repository link (i.e sharepoint) |
|  |

TABLE OF CONTENT

1 Purpose of this document 4

1.1 Context 4

1.2 Objectives 4

1.3 Pre-required 4

1.4 References 5

2 Description of PostGreSQL Zabbix package 6

2.1 check\_postgres script 6

2.2 How to run the Perl script? 8

2.3 Creation of package 9

3 Configuration in Zabbix 10

3.1 Creating PGS folder on target server 10

3.2 userparameter file 10

3.3 Affect the Template 10

# Purpose of this document

## Context

This document is intended to describe step by step installation procedure of a Zabbix PostGreSQL template.

## Objectives

How to use this template and package for monitoring

## Pre-required

This section will present mandatory pre-requisites.

O.S:

Target : host

|  |  |
| --- | --- |
| **Operating system** | **Version** |
| RedHat Linux | 5.x / 6.x / 7.x |
| CentOS | 5.x / 6.x / 7.x |

Database PostGres:

Target : Host database engine

|  |  |
| --- | --- |
| **PostreSQL** | **Version** |
| **PostreSQL** | 9.x / 10.x |

Important note about the PostGreSQL 10.x version:

It’s supported with a new version of the Perl script (check\_postgres.pl version 2.24.0). This file will be Included in the new package “zbx\_PosgreSQL\_PGH\_Kit\_v6.tar.gz”.

Zabbix Package:

Target: Repository of Proxy parking

|  |  |
| --- | --- |
| **Template package** | **Version** |
| zbx\_PosgreSQL\_PGH\_Kit\_vX.tar.gz | X=5 … could be 6 in the future. |

Specific configuration: sudo

Target : host

it is a Linux system prerequisite.

All Linux servers normally have this sudo configuration deployed.

For information only:

Possible test: (for debug use only)

Execute this command on the target host, using “walle user

**sudo -l | grep postgres**

You must have the following result with the CORE type hosts:

**walle ALL=(postgres)NOPASSWD:/etc/zabbix/zabbix\_agentd.d/PGS/zbx\_PG\_discover.sh**

Database user

Need to create a walle user :

SGDB PostGreSQL:

An account “walle” must be created with the appropriate role (DBA action).

SYSTEM: (Linux monitored host)

sudo rights

The script zbx\_PG\_discover.sh needs to read information in the PostGres configuration file.

walle ALL=(postgres)NOPASSWD: /etc/zabbix/zabbix\_agentd.d/PGS/zbx\_PG\_discover.sh

OTHERS:

perl script named check\_postgres (version 2.22.0)

This perl script needs:

**perl516-perl-Data-Dumper.x86\_64 (and all the dependencies)**

Examples:

* perl-PathTools-3.40-5.el7.x86\_64
* perl-threads-shared-1.43-6.el7.x86\_64
* perl-Getopt-Long-2.40-3.el7.noarch
* perl516-runtime-1.1-3.el7.x86\_64
* perl516-perl-Pod-Perldoc-3.20-4.el7.noarch
* perl516-perl-Pod-Usage-1.63-2.el7.noarch
* perl516-perl-PathTools-3.40-3.el7.x86\_64
* perl516-perl-Exporter-5.66-19.el7.noarch
* perl516-perl-constant-1.23-19.el7.noarch
* perl516-perl-Pod-Simple-3.28-293.el7.noarch
* perl-podlators-2.5.1-3.el7.noarch
* perl-Encode-2.51-7.el7.x86\_64
* perl-Time-HiRes-1.9725-3.el7.x86\_64
* perl-Socket-2.010-4.el7.x86\_64
* perl-Storable-2.45-3.el7.x86\_64
* perl-Scalar-List-Utils-1.27-248.el7.x86\_64
* perl-File-Path-2.09-2.el7.noarch
* perl-threads-1.87-4.el7.x86\_64
* perl-Pod-Simple-3.28-4.el7.noarch
* perl-5.16.3-292.el7.x86\_64
* perl-TermReadKey-2.30-20.el7.x86\_64
* perl516-perl-HTTP-Tiny-0.017-19.el7.noarch
* perl516-perl-podlators-2.5.3-2.el7.noarch
* perl516-perl-Pod-Escapes-1.04-19.el7.noarch
* perl516-perl-Encode-2.57-1.el7.x86\_64
* perl516-perl-macros-5.16.3-19.el7.x86\_64
* perl516-perl-Scalar-List-Utils-1.31-2.el7.x86\_64
* perl516-perl-Carp-1.32-3.el7.noarch
* perl516-perl-File-Path-2.08.01-19.el7.noarch
* perl516-perl-Socket-2.013-2.el7.x86\_64
* perl516-perl-File-Temp-0.22-19.el7.noarch
* perl516-perl-threads-shared-1.43-3.el7.x86\_64
* perl516-perl-libs-5.16.3-19.el7.x86\_64
* perl516-perl-Getopt-Long-2.38-19.el7.noarch
* perl516-perl-Data-Dumper-2.145-5.el7.x86\_64
* perl-HTTP-Tiny-0.033-3.el7.noarch
* perl-Pod-Perldoc-3.20-4.el7.noarch
* perl-Text-ParseWords-3.29-4.el7.noarch
* perl-Pod-Usage-1.63-3.el7.noarch
* perl-libs-5.16.3-292.el7.x86\_64
* perl-Exporter-5.68-3.el7.noarch
* perl-Time-Local-1.2300-2.el7.noarch
* perl-Carp-1.26-244.el7.noarch
* perl-File-Temp-0.23.01-3.el7.noarch
* perl-Filter-1.49-3.el7.x86\_64
* perl-Error-0.17020-2.el7.noarch
* perl-Git-1.8.3.1-13.el7.noarch
* perl516-perl-parent-0.228-1.el7.noarch
* perl516-perl-Text-ParseWords-3.27-19.el7.noarch
* perl516-perl-Storable-2.34-19.el7.x86\_64
* perl516-perl-Filter-1.49-4.el7.x86\_64
* perl516-perl-Time-Local-1.2000-19.el7.noarch
* perl516-perl-threads-1.89-2.el7.x86\_64
* perl516-perl-5.16.3-19.el7.x86\_64
* perl-parent-0.225-244.el7.noarch
* perl-Pod-Escapes-1.04-292.el7.noarch
* perl-macros-5.16.3-292.el7.x86\_64
* perl-constant-1.27-2.el7.noarch

## References

| Reference | Item |
| --- | --- |
| check\_postgres | https://github.com/bucardo/check\_postgres |
|  | http://www.bucardo.org/check\_postgres/ |
|  |  |

# Description of PostGreSQL Zabbix package

## check\_postgres script

“check\_postgres.pl” is a Perl script that runs many different tests against one or more Postgres databases. It uses the psql program to gather the information, and outputs the results in one of three formats: Nagios, MRTG, or simple.

The S.G. group uses it for many years

This script has a lot of options to collect monitoring metrics.

**Common metrics collection options:**

* archive\_ready => [1, 'Check the number of WAL files ready in the pg\_xlog/archive\_status'],
* autovac\_freeze => [1, 'Checks how close databases are to autovacuum\_freeze\_max\_age.'],
* backends => [1, 'Number of connections, compared to max\_connections.'],
* bloat => [0, 'Check for table and index bloat.'],
* checkpoint => [1, 'Checks how long since the last checkpoint'],
* cluster\_id => [1, 'Checks the Database System Identifier'],
* commitratio => [0, 'Report if the commit ratio of a database is too low.'],
* connection => [0, 'Simple connection check.'],
* custom\_query => [0, 'Run a custom query.'],
* database\_size => [0, 'Report if a database is too big.'],
* dbstats => [1, 'Returns stats from pg\_stat\_database: Cacti output only'],
* disabled\_triggers => [0, 'Check if any triggers are disabled'],
* disk\_space => [1, 'Checks space of local disks Postgres is using.'],
* fsm\_pages => [1, 'Checks percentage of pages used in free space map.'],
* fsm\_relations => [1, 'Checks percentage of relations used in free space map.'],
* hitratio => [0, 'Report if the hit ratio of a database is too low.'],
* hot\_standby\_delay => [1, 'Check the replication delay in hot standby setup'],
* index\_size => [0, 'Checks the size of indexes only.'],
* table\_size => [0, 'Checks the size of tables only.'],
* relation\_size => [0, 'Checks the size of tables and indexes.'],
* last\_analyze => [0, 'Check the maximum time in seconds since any one table has been analyzed.'],
* last\_vacuum => [0, 'Check the maximum time in seconds since any one table has been vacuumed.'],
* last\_autoanalyze => [0, 'Check the maximum time in seconds since any one table has been autoanalyzed.'],
* last\_autovacuum => [0, 'Check the maximum time in seconds since any one table has been autovacuumed.'],
* listener => [0, 'Checks for specific listeners.'],
* locks => [0, 'Checks the number of locks.'],
* logfile => [1, 'Checks that the logfile is being written to correctly.'],
* new\_version\_bc => [0, 'Checks if a newer version of Bucardo is available.'],
* new\_version\_box => [0, 'Checks if a newer version of boxinfo is available.'],
* new\_version\_cp => [0, 'Checks if a newer version of check\_postgres.pl is available.'],
* new\_version\_pg => [0, 'Checks if a newer version of Postgres is available.'],
* new\_version\_tnm => [0, 'Checks if a newer version of tail\_n\_mail is available.'],
* pgb\_pool\_cl\_active => [1, 'Check the number of active clients in each pgbouncer pool.'],
* pgb\_pool\_cl\_waiting => [1, 'Check the number of waiting clients in each pgbouncer pool.'],
* pgb\_pool\_sv\_active => [1, 'Check the number of active server connections in each pgbouncer pool.'],
* pgb\_pool\_sv\_idle => [1, 'Check the number of idle server connections in each pgbouncer pool.'],
* pgb\_pool\_sv\_used => [1, 'Check the number of used server connections in each pgbouncer pool.'],
* pgb\_pool\_sv\_tested => [1, 'Check the number of tested server connections in each pgbouncer pool.'],
* pgb\_pool\_sv\_login => [1, 'Check the number of login server connections in each pgbouncer pool.'],
* pgb\_pool\_maxwait => [1, 'Check the current maximum wait time for client connections in pgbouncer pools.'],
* pgbouncer\_backends => [0, 'Check how many clients are connected to pgbouncer compared to max\_client\_conn.'],
* pgbouncer\_checksum => [0, 'Check that no pgbouncer settings have changed since the last check.'],
* pgagent\_jobs => [0, 'Check for no failed pgAgent jobs within a specified period of time.'],
* prepared\_txns => [1, 'Checks number and age of prepared transactions.'],
* query\_runtime => [0, 'Check how long a specific query takes to run.'],
* query\_time => [1, 'Checks the maximum running time of current queries.'],
* replicate\_row => [0, 'Verify a simple update gets replicated to another server.'],
* same\_schema => [0, 'Verify that two databases have the exact same tables, columns, etc.'],
* sequence => [0, 'Checks remaining calls left in sequences.'],
* settings\_checksum => [0, 'Check that no settings have changed since the last check.'],
* slony\_status => [1, 'Ensure Slony is up to date via sl\_status.'],
* timesync => [0, 'Compare database time to local system time.'],
* txn\_idle => [1, 'Checks the maximum "idle in transaction" time.'],
* txn\_time => [1, 'Checks the maximum open transaction time.'],
* txn\_wraparound => [1, 'See how close databases are getting to transaction ID wraparound.'],
* version => [1, 'Check for proper Postgres version.'],
* wal\_files => [1, 'Check the number of WAL files in the pg\_xlog directory']

**Common connection options:**

* -H, --host=NAME hostname(s) to connect to; defaults to none (Unix socket)
* -p, --port=NUM port(s) to connect to; defaults to $opt{defaultport}.
* -db, --dbname=NAME database name(s) to connect to; defaults to 'postgres' or 'template1'
* -u --dbuser=NAME database user(s) to connect as; defaults to '$opt{defaultuser}'
* --dbpass=PASS database password(s); use a .pgpass file instead when possible
* --dbservice=NAME service name to use inside of pg\_service.conf

**Note:**

For a complete list of options and full documentation, view the manual.

./check\_postgres.pl --man

## How to run the Perl script?

The Zabbix monitoring tool can run scripts with the Zabbix technique named “user parameter”.

We use two files to be able to run the “check\_postgres.pl” Perl script.

* zbx\_userparameter\_postgresql.conf (for the Zabbix agent)
* zbx\_postgresql.sh (bash file)

Examples of configuration (for the “commit ratio” metric):

(parts of files)

From zbx\_userparameter\_postgresql.conf

UserParameter=zbx.postgre.commitratio[\*],/etc/zabbix/zabbix\_agentd.d/PGS/zbx\_postgresql.sh $1 $2 $3 $4 commitratio

From zbx\_postgresql.sh

commitratio)

perl ${TOOLPATH}/check\_postgres.pl --dbuser=${DBUSER} --dbname=postgres --port=${PGSQL\_PORT} --action commitratio --output=simple

;;

Examples of files:



## Package

Package name: zbx\_PosgreSQL\_PGH\_Kit\_vXX.tar.gz

Example: zbx\_PosgreSQL\_PGH\_Kit\_v2A

**Content:**

* zbx\_userparameter\_postgresql

/PGS (folder)

* check\_postgres.pl
* zbx\_PG\_discover
* zbx\_pghoard.ksh
* zbx\_postgresql.sh

## Creating PGS folder on target server

* Connecting to the server to monitor
* Create a new folder “PGS” in /etc/zabbix/zabbix\_agentd.d/

**mkdir jboss**

* Move the check\_jmx package in this folder

Ex: **mv /tmp/ check\_jmx.sh /etc/zabbix/zabbix\_agentd.d/jboss**

* Assign executive permission to check\_jmx.sh

**chmod +x check\_jmx.sh**

## userparameter file

**Manual:**

Go to /etc/zabbix/zabbix\_agentd.d/

Create/Copy the zbx\_userparameter\_postgresql.conf file.

* Restart Zabbix agent

Or:

* Copy the package zbx\_PosgreSQL\_PGH\_Kit\_vXX.tar.gz in /etc/zabbix/zabbix\_agentd.d/
* Untar the file : tar xzvf zbx\_PosgreSQL\_PGH\_Kit\_vXX.tar.gz
* Restart the Zabbix agent

## Affect the Template

Two templates are available:

Names of template :

SG\_POSTGRESQL

and

SG\_POSTGRESQL\_PGHOARD

The second template is for PGAAS hosts with pghoard backup.

Connect to Zabbix frontend , select your host .

Check if this host is enabled and available. if everything is , affect the PostGreSQL template to this host .

After 10 minutes, check if data are collected.

If there is a problem, check “Problem”.