

WCS to Zabbix integration

- [Common Zabbix agent setup overview](#)
- [Main WCS parameters](#)
 - [Configuration](#)
 - [Scripts](#)
 - [WCS statistics receiving](#)
 - [TCP connections statistics receiving](#)
 - [Garbage collector \(GC\) statistics receiving](#)
 - [WCS Websocket response checking](#)
 - [WCS process execution checking](#)
- [Zabbix template example to get WCS data](#)

[Zabbix](#) is an open source monitoring system under GPL v2. A couple of scripts launched by Zabbix agent on monitored server can be used to send WCS parameters to Zabbix server.

Below, WCS metrics collection and padding is described without paying attention to operating system and hardware metrics.

Common Zabbix agent setup overview

The configuration files for WCS parameters to collect and send to Zabbix server should be placed to `/etc/zabbix/zabbix_agent.d` folder

Script files to collect the metrics should be placed to `/etc/zabbix/scripts.d` folder

Main WCS parameters

Configuration

Main WCS parameters to collect and send should be listed in `webcallserver.conf` file

```
UserParameter=wcs.all,/etc/zabbix/scripts.d/get_stat.py
UserParameter=tcp.all,/etc/zabbix/scripts.d/tcp_status.sh
UserParameter=gc.type_gc,/etc/zabbix/scripts.d/gc.sh type_gc
UserParameter=gc.heap_size,/etc/zabbix/scripts.d/gc.sh heap_size
UserParameter=gc.used_mark_start,/etc/zabbix/scripts.d/gc.sh used_mark_start
UserParameter=gc.used_relocate_end,/etc/zabbix/scripts.d/gc.sh
```

```
used_relocate_end
UserParameter=gc.pause_sum,/etc/zabbix/scripts.d/gc.sh pause_sum
UserParameter=websocket.check,/etc/zabbix/scripts.d/websocket_check.sh
UserParameter=wcs.check,/etc/zabbix/scripts.d/wcs_check.sh
```

Where

- `wcs.all` - WCS [statistics](#) data
- `tcp.all` - TCP statistics data
- `gc.type_gc` - Java garbage collector (GC) type
- `gc.heap_size` - Java heap current size
- `gc.used_mark_start` - Java heap used by the moment of GC start
- `gc.used_relocate_end` - Java heap used by the moment of GC stop
- `gc.pause_sum` - GC duration (JVM is paused during this time)
- `websocket.check` - WCS Websocket response check
- `wcs.check` - WCS process execution on the node check

Scripts

WCS statistics receiving

WCS statistics parameters are collected by `get_stat.py` script

get_stat.py [Expand source](#)

```
#!/usr/bin/env python

import json,requests,socket
from pyzabbix import ZabbixSender, ZabbixMetric

sender_host = socket.gethostname()

r = requests.get("http://localhost:8081/?action=stat")
res = r.text.split("\n")

sender = ZabbixSender('192.168.1.179',use_config=True,chunk_size=250)

metrics = []

for item in res:
    if not item.startswith("---") and item != "":
        key = item.split("=")[0]
        value = item.split("=")[1]
        if key.startswith("native_resources"):
            pass
        else:
            m = ZabbixMetric(sender_host, 'wcs[' + key + ']', value)
```

```
metrics.append(m)

sender.send(metrics)

print(1)
```

The script parses statistics page <http://localhost:8081/?action=stat> and sends parameters to Zabbix server by address defined in the script, excluding native_resources section

On the Zabbix side parameter values can be retrieved by the following keys:

```
wcs[connections]
wcs[wcs_version]
```

TCP connections statistics receiving

TCP connections statistics is collected by tcp_status.sh script

tcp_status.sh Expand source

```
#!/bin/bash

HOST=`/bin/hostname`

/usr/sbin/ss -ant | awk "{if (NR>1) {state[\\$1]++} END {host = \\\"${HOST}\\\"; \
\
  for (i in state) {s=i; \
    sub (/ESTAB/, \\\"establ\\\", s); sub (/LISTEN/, \\\"listen\\\", s); sub (/SYN-SENT/, \\\"synsent\\\", s); \
    sub (/SYN-RECV/, \\\"synrecv\\\", s); sub (/FIN-WAIT-1/, \\\"finw1\\\", s); sub (/FIN-WAIT-2/, \\\"finw2\\\", s); \
    sub (/CLOSE-WAIT/, \\\"closew\\\", s); sub (/TIME-WAIT/, \\\"timew\\\", s); print host, \\\"tcp.\\\"s, state[i]}\" \
  | /usr/bin/zabbix_sender -c /etc/zabbix/zabbix_agentd.conf -s ${HOST} --port '10051' -i - >/dev/null 2>&1
echo \"1\"
exit 0
```

The script collect current TCP connections state and sends to Zabbix server

Garbage collector (GC) statistics receiving

JVM garbage collector (GC) statistics is collected by gc.sh script

gc.sh Expand source

```
#!/bin/bash

TYPE=$1

WCS_HOME="/usr/local/FlashphonerWebCallServer"
```

```

LAST_LOG=$(ls -t ${WCS_HOME}/logs/ | grep gc-core | head -1)
LOG="${WCS_HOME}/logs/${LAST_LOG}"
JAVA_VER=$(java -version 2>&1 | head -n 1 | awk -F ' ' '{print $2}')

TYPE_GC="$(grep -Pv '^(#|$)' ${WCS_HOME}/conf/wcs-core.properties | grep -oE
'ConcMarkSweepGC|ZGC')"

if [[ -n "TYPE_GC" ]]; then
    if [[ $TYPE == "type_gc" ]]; then
        echo $TYPE_GC
    fi
else
    echo "Garbage Collector configuration not found in WCS core.properties"
    exit 1
fi

# Used OpenJDK 1.x or Java x; GC - ConcMarkSweepGC (only)
if [[ $JAVA_VER != "1"[0-9]* ]]; then

# 2019-08-29T03:48:51.481+0700: 153426.640: [GC (Allocation Failure) 2019-08-
29T03:48:51.481+0700: 153426.640: [ParNew: 34153K->384K(38080K), 0.0045083
secs] 52756K->18988K(122752K), 0.0047446 secs] [Times: user=0.01 sys=0.00,
real=0.01 secs]

    if [[ $TYPE == "heap_size" ]]; then
        grep 'Allocation Failure' $LOG | tail -1 | awk -F'->' '{print $3}' |
sed -rn 's/([0-9]+)K\(([0-9]+)K\), ([0-9]+\.[0-9]+)\.*/\2/p' | awk '{printf
"%i\n", $1 * 1024}'

        elif [[ $TYPE == "used_mark_start" ]]; then
            grep 'Allocation Failure' $LOG | tail -1 | awk -F'->' '{print $2}' |
sed -rn 's/.* ([0-9]+)K$/\1/p' | awk '{printf "%i\n", $1 * 1024}'

            elif [[ $TYPE == "used_relocate_end" ]]; then
                grep 'Allocation Failure' $LOG | tail -1 | awk -F'->' '{print $3}' |
sed -rn 's/([0-9]+)K\(([0-9]+)K\), ([0-9]+\.[0-9]+)\.*/\1/p' | awk '{printf
"%i\n", $1 * 1024}'

                elif [[ $TYPE == "pause_sum" ]]; then
                    grep 'Allocation Failure' $LOG | tail -1 | awk -F'->' '{print $3}' |
sed -rn 's/([0-9]+)K\(([0-9]+)K\), ([0-9]+\.[0-9]+)\.*/\3/p' | tr , . | awk
'{printf "%f\n", $1 * 1000 }'
                    fi

# Used OpenJDK 1x or Java 1x; GC - ConcMarkSweepGC, ZGC
elif [[ $JAVA_VER == "1"[0-9]* ]]; then

    if [[ $TYPE_GC == "ConcMarkSweepGC" ]]; then

# [26002.922s][info][gc] GC(133) Pause Young (Allocation Failure) 101M-
>21M(1014M) 4.239ms
        if [[ $TYPE == "heap_size" ]]; then
            grep 'Allocation Failure' $LOG | tail -1 | awk '{print $7}' | awk
-F'->' '{print $2}' | sed -rn 's/[0-9]+M\(([0-9]+)M\).*/\1/p' | awk '{printf
"%i\n", $1 * 1024 * 1024}'

            elif [[ $TYPE == "used_mark_start" ]]; then
                grep 'Allocation Failure' $LOG | tail -1 | awk '{print $7}' | sed

```

```

-rn 's/([0-9]+).*/\1/p' | awk '{printf "%i\n", $1 * 1024 * 1024}'

    elif [[ $TYPE == "used_relocate_end" ]]; then
        grep 'Allocation Failure' $LOG | tail -1 | awk '{print $7}' | awk
-F'->' '{print $2}' | sed -rn 's/([0-9]+).*/\1/p' | awk '{printf "%i\n", $1 *
1024 * 1024}'

    elif [[ $TYPE == "pause_sum" ]]; then
        grep 'Allocation Failure' $LOG | tail -1 | awk '{print $8}' | sed
's/ms$//'
    fi
fi

if [[ $TYPE_GC == "ZGC" ]]; then

# [6.960s][info][gc,heap      ] GC(1) Capacity:      4096M (100%)      4096M
(100%)      4096M (100%)      4096M (100%)      4096M (100%)      4096M
(100%)
    if [[ $TYPE == "heap_size" ]]; then
        grep 'GC(.*)Capacity' $LOG | tail -1 | awk -F 'Capacity:'
'{print $2}' | sed -rn 's/^[ ]*([0-9]+).*/\1/p' | awk '{printf "%i\n", $1 *
1024 * 1024}'

# 6.960s][info][gc          ] GC(1) Garbage Collection (Metadata GC
Threshold) 186M(5%)->70M(2%)
    elif [[ $TYPE == "used_mark_start" ]]; then
        grep 'Garbage Collection (.*) ' $LOG | tail -1 | awk -F
'Allocation Rate)|Proactive)|Warmup)|Threshold)' '{print $2}' | sed 's/^\ *
([0-9]*\).*/\1/' | awk '{printf "%i\n", $1 * 1024 * 1024}'

    elif [[ $TYPE == "used_relocate_end" ]]; then
        grep 'Garbage Collection (.*) ' $LOG | tail -1 | awk -F
'Allocation Rate)|Proactive)|Warmup)|Threshold)' '{print $2}' | awk -F'->'
'{print $2}' | sed -rn 's/([0-9]+).*/\1/p' | awk '{printf "%i\n", $1 * 1024 *
1024}'

    elif [[ $TYPE == "pause_mark_start" ]]; then
        grep '.*GC.*Pause Mark Start' $LOG | tail -1 | awk -F 'Pause Mark
Start ' '{print $2}' | sed 's/ms$//'

    elif [[ $TYPE == "pause_mark_end" ]]; then
        grep '.*GC.*Pause Mark End' $LOG | tail -1 | awk -F 'Pause Mark
End ' '{print $2}' | sed 's/ms$//'

    elif [[ $TYPE == "pause_relocate_start" ]]; then
        grep '.*GC.*Pause Relocate Start' $LOG | tail -1 | awk -F 'Pause
Relocate Start ' '{print $2}' | sed 's/ms$//'

    elif [[ $TYPE == "pause_sum" ]]; then
        grep '.*GC.*Pause' $LOG | awk -F 'Pause Mark Start|End|Relocate
Start' '{print $2}' | tail -3 | sed 's/ms$//' | awk '{a=$1;
getline;b=$1;getline;c=$1;getline;t=a+b+c;print t}'

    fi
fi
fi

```

The script return one of the following parameters:

- gc.type_gc - Java garbage collector (GC) type
- gc.heap_size - Java heap current size
- gc.used_mark_start - Java heap used by the moment of GC start
- gc.used_relocate_end - Java heap used by the moment of GC stop
- gc.pause_sum - GC duration (JVM is paused during this time)

Zabbix agent sends those data to Zabbix server

WCS Websocket response checking

WCS Websocket response is checked by websocket_check.sh script

websocket_check.sh [Expand source](#)

```
#!/bin/bash

WCS_HOME="/usr/local/FlashphonerWebCallServer"

WSS_ADDRESS="$(grep -Pv '^(#|$)' ${WCS_HOME}/conf/flashphoner.properties |
grep -E 'wss.address' | awk -F '=' '{print $2}' | sed 's/^[ \t]*//;s/[ \t]*$//')"
[ -z $WSS_ADDRESS ] && WSS_ADDRESS='0.0.0.0'

WSS_PORT="$(grep -Pv '^(#|$)' ${WCS_HOME}/conf/flashphoner.properties | grep
-E 'wss.port' | awk -F '=' '{print $2}' | sed 's/^[ \t]*//;s/[ \t]*$//')"
[ -z $WSS_PORT ] && WSS_PORT='8443'

curl --connect-timeout 30 --insecure --silent --include \
--no-buffer \
--header "Connection: Upgrade" \
--header "Upgrade: websocket" \
--header "Host: $WSS_ADDRESS:$WSS_PORT" \
--header "Origin: https://$WSS_ADDRESS:$WSS_PORT" \
--header "Sec-WebSocket-Key: SGVsbG8sIHdvcmxkIQ==" \
--header "Sec-WebSocket-Version: 13" \
https://$WSS_ADDRESS:$WSS_PORT/ | grep -q "HTTP/1.1" && [[ $? == 0 ]] && echo
'1' || echo '0'
```

The script returns one of the following values:

- 1 - WCS responds by Websocket
- 0 - WCS does not respond by Websocket

Zabbix agent sends those data to Zabbix server

WCS process execution checking

WCS process execution is checked by wcs_check.sh script

wcs_check.sh Expand source

```
#!/bin/bash

PID="$(pgrep -f 'com.flashphoner.server.Server' | grep -v bash)"
[ -n "$PID" ] && echo "1" || echo "0"
```

The script returns one of the following values:

- 1 - WCS process is executing on node
- 0 - WCS process is not executing on node

Zabbix agent sends those data to Zabbix server

Zabbix template example to get WCS data

Zabbix template example to get data from agent on WCS server can be downloaded here

[zbx_export_templates.xml](#)

Attachments:

■ [zbx_export_templates.xml](#) (text/xml)