

3 Passive and active agent checks

Overview

This section provides details on passive and active checks performed by [Zabbix agent](#).

Zabbix uses a JSON based communication protocol for communicating with Zabbix agent.

Passive checks

A passive check is a simple data request. Zabbix server or proxy asks for some data (for example, CPU load) and Zabbix agent sends back the result to the server.

Server request

For definition of header and data length please refer to [protocol details](#).

```
<item key>
```

Agent response

```
<DATA>[\0<ERROR>]
```

Above, the part in square brackets is optional and is only sent for not supported items.

For example, for supported items:

1. Server opens a TCP connection
2. Server sends **<HEADER><DATALEN>agent.ping**
3. Agent reads the request and responds with **<HEADER><DATALEN>1**
4. Server processes data to get the value, '1' in our case
5. TCP connection is closed

For not supported items:

1. Server opens a TCP connection
2. Server sends **<HEADER><DATALEN>vfs.fs.size[/nono]**
3. Agent reads the request and responds with
**<HEADER><DATALEN>ZBX_NOTSUPPORTED\0Cannot obtain filesystem information:
[2] No such file or directory**
4. Server processes data, changes item state to not supported with the specified error message
5. TCP connection is closed

Active checks

Active checks require more complex processing. The agent must first retrieve from the server(s) a list of items for independent processing.

The servers to get the active checks from are listed in the 'ServerActive' parameter of the agent [configuration file](#). The frequency of asking for these checks is set by the 'RefreshActiveChecks' parameter in the same configuration file. However, if refreshing active checks fails, it is retried after hardcoded 60 seconds.

The agent then periodically sends the new values to the server(s).

Getting the list of items

Agent request

```
{
  "request": "active checks",
  "host": "<hostname>"
}
```

Server response

```
{
  "response": "success",
  "data": [
    {
      "key": "log[/home/zabbix/logs/zabbix_agentd.log]",
      "delay": 30,
      "lastlogsize": 0,
      "mtime": 0
    },
    {
      "key": "agent.version",
      "delay": 600,
      "lastlogsize": 0,
      "mtime": 0
    },
    {
      "key": "vfs.fs.size[/nono]",
      "delay": 600,
      "lastlogsize": 0,
      "mtime": 0
    }
  ]
}
```

The server must respond with success. For each returned item, all properties **key**, **delay**, **lastlogsize**

and **mtime** must exist, regardless of whether item is a log item or not.

For example:

1. Agent opens a TCP connection
2. Agent asks for the list of checks
3. Server responds with a list of items (item key, delay)
4. Agent parses the response
5. TCP connection is closed
6. Agent starts periodical collection of data

Note that (sensitive) configuration data may become available to parties having access to the Zabbix server trapper port when using an active check. This is possible because anyone may pretend to be an active agent and request item configuration data; authentication does not take place unless you use [encryption](#) options.

Sending in collected data

Agent sends

```
{
  "request": "agent data",
  "session": "12345678901234567890123456789012",
  "data": [
    {
      "host": "<hostname>",
      "key": "agent.version",
      "value": "2.4.0",
      "id": 1,
      "clock": 1400675595,
      "ns": 76808644
    },
    {
      "host": "<hostname>",
      "key": "log[/home/zabbix/logs/zabbix_agentd.log]",
      "lastlogsize": 112,
      "value": " 19845:20140621:141708.521 Starting Zabbix Agent
[<hostname>]. Zabbix 2.4.0 (revision 50000).",
      "id": 2,
      "clock": 1400675595,
      "ns": 77053975
    },
    {
      "host": "<hostname>",
      "key": "vfs.fs.size[/nono]",
      "state": 1,
      "value": "Cannot obtain filesystem information: [2] No such file
or directory",
      "id": 3,
      "clock": 1400675595,

```

```
      "ns": 78154128
    },
    "clock": 1400675595,
    "ns": 78211329
  }
```

A virtual ID is assigned to each value. Value ID is a simple ascending counter, unique within one data session (identified by the session token). This ID is used to discard duplicate values that might be sent in poor connectivity environments.

Server response

```
{
  "response": "success",
  "info": "processed: 3; failed: 0; total: 3; seconds spent: 0.003534"
}
```

If sending of some values fails on the server (for example, because host or item has been disabled or deleted), agent will not retry sending of those values.

For example:

1. Agent opens a TCP connection
2. Agent sends a list of values
3. Server processes the data and sends the status back
4. TCP connection is closed

Note how in the example above the not supported status for `vfs.fs.size[/nono]` is indicated by the "state" value of 1 and the error message in "value" property.

Error message will be trimmed to 2048 symbols on server side.

Older XML protocol

Zabbix will take up to 16 MB of XML Base64-encoded data, but a single decoded value should be no longer than 64 KB otherwise it will be truncated to 64 KB while decoding.

From: <https://www.zabbix.com/documentation/4.0/> - **Zabbix Documentation 4.0**

Permanent link: <https://www.zabbix.com/documentation/4.0/manual/appendix/items/activepassive>

Last update: **2018/12/07 11:36**

