Integrating Zabbix with alerting and ITSM systems the easy way

Arturs Lontons
ZABBIX Technical Support Engineer
Without a doubt - Zabbix is THE BEST monitoring tool out there - many monitoring approaches can be used for different infrastructures.
Zabbix - notifications

But monitoring is just the first step!

What can we do when there is a problem?
Zabbix - notifications

We define Actions which contain conditions and operations!
Media types

Media are the delivery channels used for sending notifications and alerts from Zabbix.

- Email
- SMS
- Script

And, as of 4.4:
- Webhook
React to events by notifying your team members via e-mail.

- As of 4.2 the message can either be HTML or Plain text
- Configure the SMTP parameters
- Create the media for your individual users
- Test it...
- ...Not by simulating a problem! Use the **Test** button! (new in 4.2)
• Simple to configure
• Requires a GSM modem attached
• Tested models provided in the documentation
• Connected via a serial port or USB
• More complex configuration in case of a virtualized environment
• Create the media for your individual users

* Name
Type SMS
* GSM modem /dev/ttyS0
Alert script

- Executed by the Zabbix server
- Used to integrate Zabbix with 3rd party systems
- Can pass Macros as parameters
- Can be written in any language
- Examples available on Zabbix share or the Integrations section
Out of the box integrations

And that was pretty much it for Zabbix 4.0! You were fully expected to create and use your own alert scripts for integration or use the community provided solutions.

Zabbix Slack AlertScript
A bash script for Slack custom alert in Zabbix. Works with Zabbix 1.8.x or greater - including 2.2, 2.4, 3.x. and 4.x!

github.com/ericoc/zabbix-slack-alertscript

Create ticket in JIRA
script for creating Jira ticket (can be used in actions)
✓ Custom Script

github.com/alexander-nesterov/zabbix_jira

share.zabbix.com/creation-ticket-in-jira
Out of the box integrations

Problem:
How can we provide portable out of the box integrations?

Solution:
Zabbix 4.2 - Support of JavaScript (in preprocessing)
Zabbix 4.4 - Webhooks and programmable logic for actions and notifications

- Since Zabbix 4.4.4:
  - Mattermost
  - Opsgenie
  - Pushover

- Since Zabbix 4.4.5:
  - Pagerduty
  - Slack

- Since Zabbix 4.4.6:
  - Discord
Webhooks

How do the webhooks actually work?

● Passes the event related parameters
● Uses the provided API token
● Specify the endpoint in the parameter
The most important part of the webhook is the script:

```javascript
function getPermalink(channelId, messageTimestamp) {
    var req = new CurlHttpRequest();
    req.AddHeader('Content-Type: application/x-www-form-urlencoded; charset=UTF-8');

    var resp = JSON.parse(req.Get('
        (0)?token={1}&channel={2}&message_ts={3}'.format(
            slack.getPermalink,
            params.bot_token,
            channelId,
            messageTimestamp
        ));

    if (req.status !== 200 || !resp.ok) {
        throw resp.error;
    }

    return resp.permalink;
}
```
Webhooks - script

- Oh, I still have to write the script though... :(

Not anymore! WE provide the script! It’s available out of the box or you can simply import it from our git repo!

### Source

- **master**
  - Zabbix / templates / media / **slack** /

### Source Details

<table>
<thead>
<tr>
<th>Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>media_slack.xml</td>
<td>A....T [DEV-1439] added unit tests for r</td>
</tr>
<tr>
<td>README.md</td>
<td>........T [ZBXNEXT-5562] fixed slack logo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>Email</td>
<td>Enabled</td>
</tr>
<tr>
<td>Mattermost</td>
<td>Webhook</td>
<td>Enabled</td>
</tr>
<tr>
<td>Opsgenie</td>
<td>Webhook</td>
<td>Enabled</td>
</tr>
<tr>
<td>PagerDuty</td>
<td>Webhook</td>
<td>Enabled</td>
</tr>
<tr>
<td>Pushover</td>
<td>Webhook</td>
<td>Enabled</td>
</tr>
<tr>
<td>Slack</td>
<td>Webhook</td>
<td>Enabled</td>
</tr>
<tr>
<td>SMS</td>
<td>SMS</td>
<td>Enabled</td>
</tr>
</tbody>
</table>
How else is this better than a custom script? It’s Portable!

export/import the media type:

Copy your API token:

- **Name**: Slack
- **Type**: Webhook

**Parameters**
- **bot_token**: xoxb-935751598966-9336538195
- **channel**: {ALERT SENDTO}
- **slack_mode**: alarm
- **slack_as_user**: true
Webhooks - additional configuration

Starting with Zabbix 5.0 you can now specify HTTP Proxy as a parameter in Webhooks (Same logic as for HTTP agent item)

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Name</th>
<th>Value</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTTPProxy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>summary</td>
<td></td>
<td>{ALERT.SUBJECT}</td>
<td>Remove</td>
</tr>
<tr>
<td>description</td>
<td></td>
<td>{ALERT.MESSAGE}</td>
<td>Remove</td>
</tr>
</tbody>
</table>
Webhooks - additional configuration

With Webhooks we can also populate our tags with data from the external system, for example:

- Channel ID
- Channel name
- Message link
- And many more
Webhooks - additional configuration

You can add a direct link to the ticket:
Webhooks - tags

How are the tag values obtained?

The returned JSON response should contain our tags:

```json
{
    "tags": {
        "__message_ts": "1582298858.001400",
        "__channel_id": "CTF2J8PBN",
        "__channel_name": "#benelux",
        "__message_link": "https://zabbix-pr49306.slack.com/archives/CTF2J8PBN/p1582298858001400"
    }
}
```
Webhooks - tags

Once again - we should use JavaScript to format our returned data accordingly:

```javascript
var resp = JSON.parse(req.Post($slack.postMessage, JSON.stringify(fields)));
if (req.Status != 200 & !resp.ok) {
    throw resp.error;
}

result.tags._message_ts = resp.ts;
result.tags._channel_id = resp.channel;
result.tags._channel_name = $params.channel;
result.tags._message_link = getPermalink(resp.channel, resp.ts);

else if ($isEventUpdate($params)) {
    fields.thread_ts = $tagValue($params.event.tags, 'message_ts');
    fields.attachments = [
        createMessage($EVENT_STATUS.UPDATE,
```
Let’s take a look at the steps needed to implement Zabbix integration with Slack WITHOUT using Webhooks - the legacy way!

As an example we will use zabbix-notify. A solution available on our community share

- Featured integration
- Uses AlertScript
- Written in Perl
Integration example - legacy approach

• Step 1: Install the required modules and dependencies

We need to make sure that *LWP* and *JSON::XS* modules are installed.

The command depends on your favorite flavor of a package manager, for example:

```
yum install perl-JSON-XS perl-libwww-perl perl-LWP-Protocol-https perl-parent
```
Integration example - legacy approach

- Step 2: Download the .tar and install it into your system

perl Makefile.PL INSTALLSITESCRIPT=/usr/local/share/zabbix/alertscripts
make test
make install

`INSTALLSITESCRIPT` should point towards your AlertScriptsPath directory, as per your Zabbix server configuration.
Integration example - legacy approach

- Step 3: Create a slack Application
Integration example - legacy approach

- Step 4: Create a slack bot user
Integration example - legacy approach

- Step 5: Install app into your workspace
Integration example - legacy approach

- Step 6: Obtain your bot token
Integration example - legacy approach

- Step 7: Create a new media type and provide the token
● Step 8: Create a new media for your users
Integration example - legacy approach

- Step 9: Create an action
Integration example - legacy approach

- Step 10: Receive your slack alerts!
Integration example - legacy approach

Let’s summarize the additional tasks we had to do to implement such an integration:

- Download and configure the script
- Install custom modules
- Test the integration manually

Some compatibility issues may also be encountered!
Integration example - legacy approach

- No guaranteed long-term support

```
zabbix-notify
```

- Possible dependency/compilation issues based on the underlying OS

```
[root@localhost ~]# yum install perl-JSON-XS
Last metadata expiration check: 0:19:10 ago on Tue 03 Mar 2020 04:29:14 AM EST.
No match for argument:
Error: Unable to find a match: perl-JSON-XS
[root@localhost ~]#
```
Integration example - Webhooks

Now, let’s look at our official Slack integration!

• Step 1: Repeat steps 3-6 to create a slack Application and obtain the bot user token!
• Step 2: Create a {$ZABBIX.URL} macro which will contain a link to your Zabbix instance
• Step 3: Import the media type
• Step 4: set bot_token to the previously created token
• Step 5: Create a media for the corresponding user
Integration example - Webhooks

And you’re done! Let’s look at our Slack alert:

**PROBLEM: Server Unreachable**

- **Host**: Linux 01 [127.0.0.1]
- **Severity**: Warning
- **Event time**: 2020.02.21 10:27:35
- **Opdata**: 1

**Trigger description**

[Open in Zabbix]
While the amount of performed steps is very similar, the integration was quite a bit more streamlined:

- No additional module or script download necessary
- The script was contained in the imported integration
- Really easy to do even for users not familiar with Zabbix configuration
- Fully done within Zabbix and Slack, no need to execute any commands on the OS!
- Guaranteed future support and additional updates!
QUESTIONS?
THANK YOU!

Arturs Lontons
ZABBIX Technical Support Engineer