Mastodon

Zabbix

Zabbix Install with Docker

Installing Zabbix Monitor with Docker

https://www.youtube.com/embed/ec2G1PeLS5k

I've been asked about Zabbix for a while now. I have covered some other solutions for monitoring equipment / services in the past. CheckMK, using Dashy Widgets, Glances, NetData, and several others are some really great options for monitoring, and as with any software, the right thing for you will depend greatly on your needs. As I started looking at Zabbix, I was contacted by Marc over at OneMarcFifty about doing a collaborative video series with him on Zabbix. He asked if I would be interested in covering the install, and he would cover some more in-depth setup of getting monitored systems enrolled, as well as setting up email alerts for anything the system finds.

How could I say, "No" to such a great opportunity? Of course I was interested. Marc does some absolutely amazing content, and I have watched his channel for a couple of years now. He has some incredibly great content on all kinds of tech topics, and his explanations are just terrific, so definitely jump over to his channel for the second part of this tutorial once you've got Zabbix up and running. You can find the video right here.

Installation

What you'll need

- A system with Docker-CE and Docker-Compose installed (we'll call this the Host Server for this tutorial)
- SSH Access to the Host Server
- Git, Curl, Wget installed on the Host Server

- About 30 Minutes of your time

Installation of Docker via a Simple Script

You can easily install Docker-CE, Docker-Compose, Portainer-CE, and NGinX Proxy manager by using this quick install script I created and maintain on Github. Just use the command:

`wget https://gitlab.com/bmcgonag/docker_installs/-/raw/main/install_docker_nproxyman.sh`

To download the script to your desired host.

Change the permissions to make the script executable:

`chmod +x ./install_docker_nproxyman.sh`

and then run the script with the command:

`./install_docker_nproxyman.sh`

When run, the script will prompt you to select your host operating system, then will ask you which bits of software you want to install.

Simply enter 'y' for each thing you want to install. In this case, you really only need to install Docker-CE and Docker-Compose. Feel free to answer 'n' to the other software options.

At some point, you may be asked for your super user (sudo) password as well.

Allow the script to complete installation.

At this point, you might want to log out and back in, as this will allow you to use the `docker` and `docker-compose` commands without the need of sudo in front of them.

Alternatively, you can try the following commands:

`newgrp`

`newgrp docker`

Now you can test your ability to run a docker command by doing

`docker ps`

If you don't get any errors, you are good to go.

Installing Zabbix

Now that we have our server software setup, we'll start on our Zabbix installation. Fortunately, Zabbix provides a nice set of ready to use docker-compose options for us. We'll clone the Zabbix-Docker repository from github, and then make a few modifications to some specific files to set our Environment Variables, and then we'll be ready to run.

First, we need to make sure we have git, curl, and wget installed on our Host Server. We can install them on Debian / Ubuntu with

```
sudo apt install git curl wget -y
```

On Fedora, CentOS, Redhat we should be able to use:

```
sudo dnf install git curl wget -y
```

On Arch, you should be able to use

```
sudo pacman -S git curl wget
```

Once those few dependencies are installed, we'll use the git command to pull down the Zabbix-Docker repository to our local machine. First, however, let's move into the "docker" folder. If you don't already have a top level folder to keep all of your docker applications in, I highly recommend you set one up, as you can then simply compress and backup the top level folder and have all of your docker applications and data backed up with one command.

```
cd docker
```

Now let's clone that repository:

```
git clone https://github.com/zabbix/zabbix-docker.git
```

This will create a folder called "zabbix-docker" with all of the files from the repository in it. We'll move into that folder with

```
cd zabbix-docker
```

In this folder you'll find many docker-compose template files. These are labeled in a way that is fairly easy to understand. Each file that has "local" in the name, means that the compose file will attempt to build new images when run, versus the files that only have "latest" and not "local" in the name. These files will pull down pre-built images from dockerhub for us to use in our system.

In my case, I chose the file called "docker-compose_v3_alpine_mysql_latest.yml". Now, you can run a file like this directly with the docker-compose command if you use the proper flags, but what I prefer to do is copy the file that I want to use to a new filed simply called "docker-compose.yml". I can then modify this file if needed, but will still have the original in place in case I need to start over cleanly. So, let's copy our selected file:

```
cp docker-compose_v3_alpine_mysql_latest.yml docker-compose.yml
```

Next, we need to set a few environment variables. These variables are super useful, especially in large projects like Zabbix, because you can set a value one time, and it is reused throughout the project. This reduces issues with misspellings, mimatched values throughout a docker-compose file, etc.

We'll move into the environment variable folder of the project with the command:

```
cd env_vars
```

In this folder, are a group of hidden files. In Linux / Unix based systems, hidden files are set by the use of a dot / period in front of the file name. In order to see these files in our directory we use the flag "a" with the "ls" command. We can also use the flag "l" to list out the file permissions, and make them list vertically down the screen. So, let's list out the files:

```
ls -al
```

We need to set a few variables. Most of these files we will leave untouched, however.

First, we'll edit the following files since we are using the mysql version of the docker-compose they provide.

- MYSQL_PASSWORD
- MYSQL_ROOT_PASSWORD
- MYSQL_USER

We don't need to change the POSTGRES_PASSWORD, POSTGRES_USER, or MYSQL_ROOT_USER values.

> NOTE: If you prefer to use the POSTGRES based dockeer-compose, you would change the POSTGRES environment variables instead of the MYSQL environment variables. Just follow the same procedure outlined below for each file.

In order to edit each file, you'll use the following command structure:

`nano <filename including the leading dot "." >`

Make the change to the value, then save using CTRL+O, then Enter to confirm, and CTRL+X to exit the nano editor.

We'll use the .MYSQL_USER as our first example:

`nano .MYSQL_USER`

You should then see this following

`zabbix`

Which you'll change to a username you want / prefer.

> NOTE: you cannot use the username "root" as it's a reserved name.

I changed mine to:

`brian`

Then save, with CTRL+O, and Enter to confirm, then CTRL+X to exit the nano editor.

Repeat this process for each of the files we will be making edits in.

Change the password values in .MYSQL_PASSWORD and .MYSQL_ROOT_PASSWORD to be long, strong passwords.

Next, we may want to make a change in a couple of the other files in this directory. I make a few modifications in the video, but they are not necessary. If you want to make changes, be certain

you understand how those changes effect the system overall.

Once, you've made changes to the necessary environment variables, we are ready to run our docker-compose.yml file.

> Since the video was made the docker-compose versions have changed a bit. In order to change the port mappings, you need to look in the file labeled .env in the main folder with all of the various compose.yaml files.

Other than potentially needing to change the port mappings inside the .env file, feel free to open it, and look through it, but there are no changes beyond nginx ports that need to be made in order to get Zabbix up and running. Again, as you get more comfortable with Zabbix, you may want to make minor modifications to this file, but only do so if you are certain you understand what affect the changes will have.

You can check to see if any of these ports are in use by running the command:

```
sudo lsof -i -P -n | grep LISTEN
```

This will show you a list of all of the ports on your host system that are in use. If you see 80, 8081, 443, or 8443, then move to the appropriate section in the .env file, and change the port **number associated to NGinX.** Use an unused port from your host. For instance, if you find that 80 is already in use on your host, then change the port for ZABBIX_WEB_NGINX_HTTP_PORT, and so on.

In my case the variable for NGinX was 80 and 443.

to use a port like 8022, I changed those lines to look like:

```
ZABBIX_WEB_NGINX_HTTP_PORT=8022
ZABBIX_WEB_NGINX_HTTPS_PORT=443
```

After you've changed all of the port mappings to use non-conflicting ports, save the file with CTRL+O, then Enter to confirm, and exit nano with CTRL+X. Just remember this change. Especially the port 80 line item. You'll need to know that port for accessing the Zabbix Web User Interface.

Now we can run:

```
docker-compose up -d && docker-compose logs -f
```

This really runs two commands: The part before the "&&" will pull down the zabbix images, and create new containers for us. The part after the "&&" will show us the logs of Zabbix starting up after the containers are started.

You will see some warnings in the logs about the limitation settings for CPU, Memory, etc. This is because docker-compose is not an orchestrator. In docker-compose v3 and later, the limit values are ignored. Such values are used in docker-swarm only.

Troubleshooting

One issue I came across during my testing of how to get all of this up and running, was that on one of my servers, my docker-compose version was not new enough. It would continuously give me an error about "profiles" not being supported. The key is to get a newer version of docker-compose installed. Depending on the version of your Linux Distribution, you may have to add a more recent repository in order to update your docker-compose version, or do more of a manual install of docker-compose. Since there are so many variables on what OS may need which setup, it's not feasible for me to try and guide you further on the topic. I ended up on docker-compose version 1.29.x and it worked without issue. ON 1.25.x I was getting the issue with the "profiles" section of the docker-compose file.

The Zabbix Web Interface

You should now be able to access the Zabbix Web Interface using your web browser of choice. You'll go to the IP address of your host machine. In my case I installed it on a machine with the IP 192.168.10.42. So in my web browser I type:

http://192.168.10.42

If you changed the 80:8080 port mapping, make sure to add the port to your IP. If I had change it from 80:8080 to 8022:8080, I would then enter the following into my browser url bar:

http://192.168.10.42:8022

Once you see the login page (be patient, as it could take a few minutes for Zabbix to come up the first time), use the username "Admin" and password "zabbix" to login.

> You should immediately navigate to the User Settings >> Profile in the left navigation bar to change your admin user password to a long, strong password.

You are now ready to begin enrolling devices into your Zabbix monitoring solution. This is a massive system with an incredible amount of power. Take your time, get to know what all it's capable of doing, and what information you can gain from it. A system like this is worth the time you'll put into making it do as much as you can.

In the video, I go through setting up one client machine. It's a manual process, but worth watching. Make sure to watch Marc's follow up video over @OneMarcFifty where he will help you unlock more of the power available in this awesome open source system.

https://www.youtube.com/embed/DFdDEf5iib4

Support My Channel and Content

Support my Channel and ongoing efforts through Patreon:
https://www.patreon.com/bePatron?u=234177