**How To Install and Configure Zabbix to Securely Monitor Remote Servers on CentOS 7**

**Introduction**

[Zabbix](http://www.zabbix.com/) is open-source monitoring software for networks and applications. It offers real-time monitoring of thousands of metrics collected from servers, virtual machines, network devices, and web applications. These metrics can help you determine the current health of your IT infrastructure and detect problems with hardware or software components before customers complain. Useful information is stored in a database so you can analyze data over time and improve the quality of provided services, or plan upgrades of your equipment.

Zabbix uses several options for collecting metrics, including agentless monitoring of user services and client-server architecture. To collect server metrics, it uses a small agent on the monitored client to gather data and send it to the Zabbix server. Zabbix supports encrypted communication between the server and connected clients, so your data is protected while it travels over insecure networks.

The Zabbix server stores its data in a relational database powered by [MySQL](https://www.mysql.com/), [PostgreSQL](https://www.postgresql.org/), or [Oracle](https://www.oracle.com/index.html). You can also store historical data in nosql databases like [Elasticsearch](https://www.elastic.co/) and [TimescaleDB](https://www.timescale.com/). Zabbix provides a web interface so you can view data and configure system settings.

You will configure two machines. One will be configured as the server, and the other as a client that you will monitor. The server will use a MySQL database to record monitoring data and use Apache to serve the web interface.

**Prerequisites**

To follow this tutorial, you will need:

* Two CentOS 7 servers set up, including a non-root user with sudo privileges and a [firewall](https://www.digitalocean.com/community/tutorials/additional-recommended-steps-for-new-centos-7-servers#configuring-a-basic-firewall).
* On one Zabbix server, you will install Zabbix;
* It will monitor your second server; this second server will be referred to as the **Zabbix Agent**.
* The Zabbix server needs Apache, MySQL, and PHP installed.
* **Note**: CentOS uses MariaDB instead of MySQL, but this will not cause any issues while following this tutorial.

**Steps:**

1. Installing the Zabbix Server
2. Setup a non-root user with sudo access
3. Setup LAMP Stack on Zabbix server
4. Secure the server with Transport Layer Security (TLS)
5. Configuring Settings for the Zabbix Web Interface
6. Installing and Configuring the Zabbix Agent
7. Adding the New Host to the Zabbix Server
8. Configuring Email Notifications
9. Generating a Test Alert

Additionally, because you will use the Zabbix server to access valuable information about your infrastructure that you would not want unauthorized users to access, it’s important that you keep your server secure by installing a TLS/SSL certificate. This is optional but **strongly encouraged**.

**Create a Sudo User**

# useradd <username>

# usermod -aG wheel <username>

# su - <username>

**How to Install Linux, Apache, MySQL, PHP (LAMP) stack On CentOS 7**

Introduction

A “LAMP” stack is a group of open-source software that is typically installed together to enable a server to host dynamic websites and web apps. This term is actually an acronym which represents the **L**inux operating system, with the **A**pache web server. The site data is stored in a **M**ySQL database (using MariaDB), and dynamic content is processed by **P**HP.

In this guide, we’ll get a LAMP stack installed on an CentOS 7 VPS. CentOS will fulfill our first requirement: a Linux operating system.

**Prerequisites**

Before you begin with this guide, you should have a separate, non-root user account set up on your server. You can learn how to do this by completing steps 1-4 in the [initial server setup for CentOS 7](https://www.digitalocean.com/community/articles/initial-server-setup-with-centos-7).

**Step One — Install Apache**

The Apache web server is currently the most popular web server in the world, which makes it a great default choice for hosting a website.

We can install Apache easily using CentOS’s package manager, yum. A package manager allows us to install most software pain-free from a repository maintained by CentOS. You can learn more about [how to use yum](https://www.digitalocean.com/community/tutorials/how-to-set-up-and-use-yum-repositories-on-a-centos-6-vps) here.

For our purposes, we can get started by typing these commands:

sudo yum install httpd

Since we are using a sudo command, these operations get executed with root privileges. It will ask you for your regular user’s password to verify your intentions.

Afterwards, your web server is installed.

Once it installs, you can start Apache on your VPS:

sudo systemctl start httpd.service

You can do a spot check right away to verify that everything went as planned by visiting your server’s public IP address in your web browser (see the note under the next heading to find out what your public IP address is if you do not have this information already):

http://your\_server\_IP\_address/

You will see the default CentOS 7 Apache web page, which is there for informational and testing purposes. It should look something like this:



If you see this page, then your web server is now correctly installed.

The last thing you will want to do is enable Apache to start on boot. Use the following command to do so:

sudo systemctl enable httpd.service

curl http://icanhazip.com

Regardless of the method you use to get your IP address, you can type it into your web browser’s address bar to get to your server.

**Step Two — Install MySQL (MariaDB)**

Now that we have our web server up and running, it is time to install MariaDB, a MySQL drop-in replacement. MariaDB is a community-developed fork of the MySQL relational database management system. Basically, it will organize and provide access to databases where our site can store information.

Again, we can use yum to acquire and install our software. This time, we’ll also install some other “helper” packages that will assist us in getting our components to communicate with each other:

sudo yum -y install mariadb-server mariadb

When the installation is complete, we need to start MariaDB with the following command:

sudo systemctl start mariadb

Now that our MySQL database is running, we want to run a simple security script that will remove some dangerous defaults and lock down access to our database system a little bit. Start the interactive script by running:

sudo mysql\_secure\_installation

The prompt will ask you for your current root password. Since you just installed MySQL, you most likely won’t have one, so leave it blank by pressing enter. Then the prompt will ask you if you want to set a root password. Go ahead and enter Y, and follow the instructions:

Enter current password for root (enter for none):

OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB

root user without the proper authorization.

New password: password

Re-enter new password: password

Password updated successfully!

Reloading privilege tables..

... Success!

For the rest of the questions, you should simply hit the “ENTER” key through each prompt to accept the default values. This will remove some sample users and databases, disable remote root logins, and load these new rules so that MySQL immediately respects the changes we have made.

The last thing you will want to do is enable MariaDB to start on boot. Use the following command to do so:

sudo systemctl enable mariadb.service

At this point, your database system is now set up and we can move on.

**Step Three — Install PHP**

PHP is the component of our setup that will process code to display dynamic content. It can run scripts, connect to our MySQL databases to get information, and hand the processed content over to our web server to display.

We can once again leverage the yum system to install our components. We’re going to include the php-mysql package as well:

sudo yum -y install php php-mysql

This should install PHP without any problems. We need to restart the Apache web server in order for it to work with PHP. You can do this by typing this:

sudo systemctl restart httpd.service

**Install PHP Modules**

To enhance the functionality of PHP, we can optionally install some additional modules.

To see the available options for PHP modules and libraries, you can type this into your system:

yum search php-

The results are all optional components that you can install. It will give you a short description for each:

php-bcmath.x86\_64 : A module for PHP applications for using the bcmath library

php-cli.x86\_64 : Command-line interface for PHP

php-common.x86\_64 : Common files for PHP

php-dba.x86\_64 : A database abstraction layer module for PHP applications

php-devel.x86\_64 : Files needed for building PHP extensions

php-embedded.x86\_64 : PHP library for embedding in applications

php-enchant.x86\_64 : Enchant spelling extension for PHP applications

php-fpm.x86\_64 : PHP FastCGI Process Manager

php-gd.x86\_64 : A module for PHP applications for using the gd graphics library

. . .

To get more information about what each module does, you can either search the internet, or you can look at the long description in the package by typing:

yum info package\_name

There will be a lot of output, with one field called Description which will have a longer explanation of the functionality that the module provides.

For example, to find out what the php-fpm module does, we could type this:

yum info php-fpm

Along with a large amount of other information, you’ll find something that looks like this:

. . .

Summary : PHP FastCGI Process Manager

URL : http://www.php.net/

License : PHP and Zend and BSD

Description : PHP-FPM (FastCGI Process Manager) is an alternative PHP FastCGI

: implementation with some additional features useful for sites of

: any size, especially busier sites.

If, after researching, you decide you would like to install a package, you can do so by using the yum install command like we have been doing for our other software.

If we decided that php-fpm is something that we need, we could type:

sudo yum -y install php-fpm

If you want to install more than one module, you can do that by listing each one, separated by a space, following the yum install command, like this:

sudo yum install package1 package2 ...

At this point, your LAMP stack is installed and configured. We should still test out our PHP though.

**Step Four — Test PHP Processing on your Web Server**

In order to test that our system is configured properly for PHP, we can create a very basic PHP script.

We will call this script info.php. In order for Apache to find the file and serve it correctly, it must be saved to a very specific directory, which is called the “web root”.

In CentOS 7, this directory is located at /var/www/html/. We can create the file at that location by typing:

sudo vi /var/www/html/info.php

This will open a blank file. We want to put the following text, which is valid PHP code, inside the file:

<?php phpinfo(); ?>

When you are finished, save and close the file.

If you are running a firewall, run the following commands to allow HTTP and HTTPS traffic:

sudo firewall-cmd --permanent --zone=public --add-service=http

sudo firewall-cmd --permanent --zone=public --add-service=https

sudo firewall-cmd --reload

Now we can test whether our web server can correctly display content generated by a PHP script. To try this out, we just have to visit this page in our web browser. You’ll need your server’s public IP address again.

The address you want to visit will be:

http://your\_server\_IP\_address/info.php

The page that you come to should look something like this:

This page basically gives you information about your server from the perspective of PHP. It is useful for debugging and to ensure that your settings are being applied correctly.

If this was successful, then your PHP is working as expected.

You probably want to remove this file after this test because it could actually give information about your server to unauthorized users. To do this, you can type this:

sudo rm /var/www/html/info.php

You can always recreate this page if you need to access the information again later.

**How To Secure Apache with Let's Encrypt on CentOS 7**

**Introduction**

[Let’s Encrypt](https://letsencrypt.org/) is a Certificate Authority (CA) that provides free certificates for [Transport Layer Security (TLS) encryption](https://www.digitalocean.com/community/tutorials/openssl-essentials-working-with-ssl-certificates-private-keys-and-csrs), thereby enabling encrypted HTTPS on web servers. It simplifies the process of creation, validation, signing, installation, and renewal of certificates by providing a software client that automates most of the steps—[Certbot](https://certbot.eff.org/).

In this tutorial, you will use Certbot to set up a TLS/SSL certificate from Let’s Encrypt on a CentOS 7 server running Apache as a web server. Additionally, you will automate the certificate renewal process using a cron job, which you can learn more about by reading [How To Use Cron To Automate Tasks On a VPS](https://www.digitalocean.com/community/tutorials/how-to-use-cron-to-automate-tasks-on-a-vps).

**Prerequisites**

In order to complete this guide, you will need:

* One CentOS 7 server set up by following [the CentOS 7 initial server setup guide](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-centos-7) with a non-root user who has sudo privileges.
* A basic firewall configured by following the [Additional Recommended Steps for New CentOS 7 Servers](https://www.digitalocean.com/community/tutorials/additional-recommended-steps-for-new-centos-7-servers#configuring-a-basic-firewall) guide.
* Apache installed on the CentOS 7 server with a virtual host configured. This tutorial will use /etc/httpd/sites-available/example.com.conf as an example.
* You should own or control the registered domain name that you wish to use the certificate with. If you do not already have a registered domain name, you may purchase one on [Namecheap](https://namecheap.com/), get one for free on [Freenom](http://www.freenom.com/en/index.html), or use the domain registrar of your choice.
* A DNS **A Record** that points your domain to the public IP address of your server. You can follow [this introduction to DigitalOcean DNS](https://www.digitalocean.com/community/tutorials/an-introduction-to-digitalocean-dns) for details on how to add them with the DigitalOcean platform. DNS A records are required because of how Let’s Encrypt validates that you own the domain it is issuing a certificate for. For example, if you want to obtain a certificate for example.com, that domain must resolve to your server for the validation process to work. Our setup will use example.com and www.example.com as the domain names, both of which will require a valid DNS record.

When you have all of these prerequisites completed, move on to install the Let’s Encrypt client software.

**Step 1 — Installing the Certbot Let’s Encrypt Client**

To use Let’s Encrypt to obtain an SSL certificate, you first need to install Certbot and [mod\_ssl](https://httpd.apache.org/docs/2.4/mod/mod_ssl.html), an Apache module that provides support for SSL v3 encryption.

The certbot package is not available through the package manager by default. You will need to enable the [EPEL](https://fedoraproject.org/wiki/EPEL) repository to install Certbot.

To add the CentOS 7 EPEL repository, run the following command:

sudo yum install epel-release

Now that you have access to the repository, install all of the required packages:

sudo yum install certbot python2-certbot-apache mod\_ssl

During the installation process you will be asked about importing a GPG key. This key will verify the authenticity of the package you are installing. To allow the installation to finish, accept the GPG key by typing y and pressing ENTER when prompted to do so.

With these services installed, you’re now ready to run Certbot and fetch your certificates.

**Step 2 — Obtaining a Certificate**

Now that Certbot is installed, you can use it to request an SSL certificate for your domain.

Using the certbot Let’s Encrypt client to generate the SSL Certificate for Apache automates many of the steps in the process. The client will automatically obtain and install a new SSL certificate that is valid for the domains you provide as parameters.

To execute the interactive installation and obtain a certificate that covers only a single domain, run the certbot command with:

sudo certbot --apache -d officialtechmayor.tk

This runs certbot with the --apache plugin and specifies the domain to configure the certificate for with the -d flag.

If you want to install a single certificate that is valid for multiple domains or subdomains, you can pass them as additional parameters to the command, tagging each new domain or subdomain with the -d flag. The first domain name in the list of parameters will be the **base** domain used by Let’s Encrypt to create the certificate. For this reason, pass the base domain name as first in the list, followed by any additional subdomains or aliases:

sudo certbot --apache -d example.com -d www.example.com

The base domain in this example is example.com.

The certbot utility can also prompt you for domain information during the certificate request procedure. To use this functionality, call certbot without any domains:

sudo certbot --apache

The program will present you with a step-by-step guide to customize your certificate options. It will ask you to provide an email address for lost key recovery and notices, and then prompt you to agree to the terms of service. If you did not specify your domains on the command line, you will be prompted for that as well. If your Virtual Host files do not specify the domain they serve explicitly using the ServerName directive, you will be asked to choose the virtual host file. In most cases, the default ssl.conf file will work.

You will also be able to choose between enabling both http and https access or forcing all requests to redirect to https. For better security, it is recommended to choose the option 2: Redirect if you do not have any special need to allow unencrypted connections. Select your choice then hit ENTER.

Output

Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

1: No redirect - Make no further changes to the webserver configuration.

2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for

new sites, or if you're confident your site works on HTTPS. You can undo this

change by editing your web server's configuration.

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Select the appropriate number [1-2] then [enter] (press 'c' to cancel):2

When the installation is successfully finished, you will see a message similar to this:

Output

IMPORTANT NOTES:

- Congratulations! Your certificate and chain have been saved at:

/etc/letsencrypt/live/example.com/fullchain.pem

Your key file has been saved at:

/etc/letsencrypt/live/example.com/privkey.pem

Your cert will expire on 2019-08-14. To obtain a new or tweaked

version of this certificate in the future, simply run certbot again

with the "certonly" option. To non-interactively renew \*all\* of

your certificates, run "certbot renew"

- If you like Certbot, please consider supporting our work by:

Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate

Donating to EFF: https://eff.org/donate-le

The generated certificate files will be available within a subdirectory named after your base domain in the /etc/letsencrypt/live directory.

Now that your certificates are downloaded, installed, and loaded, you can check your SSL certificate status to make sure that everything is working.

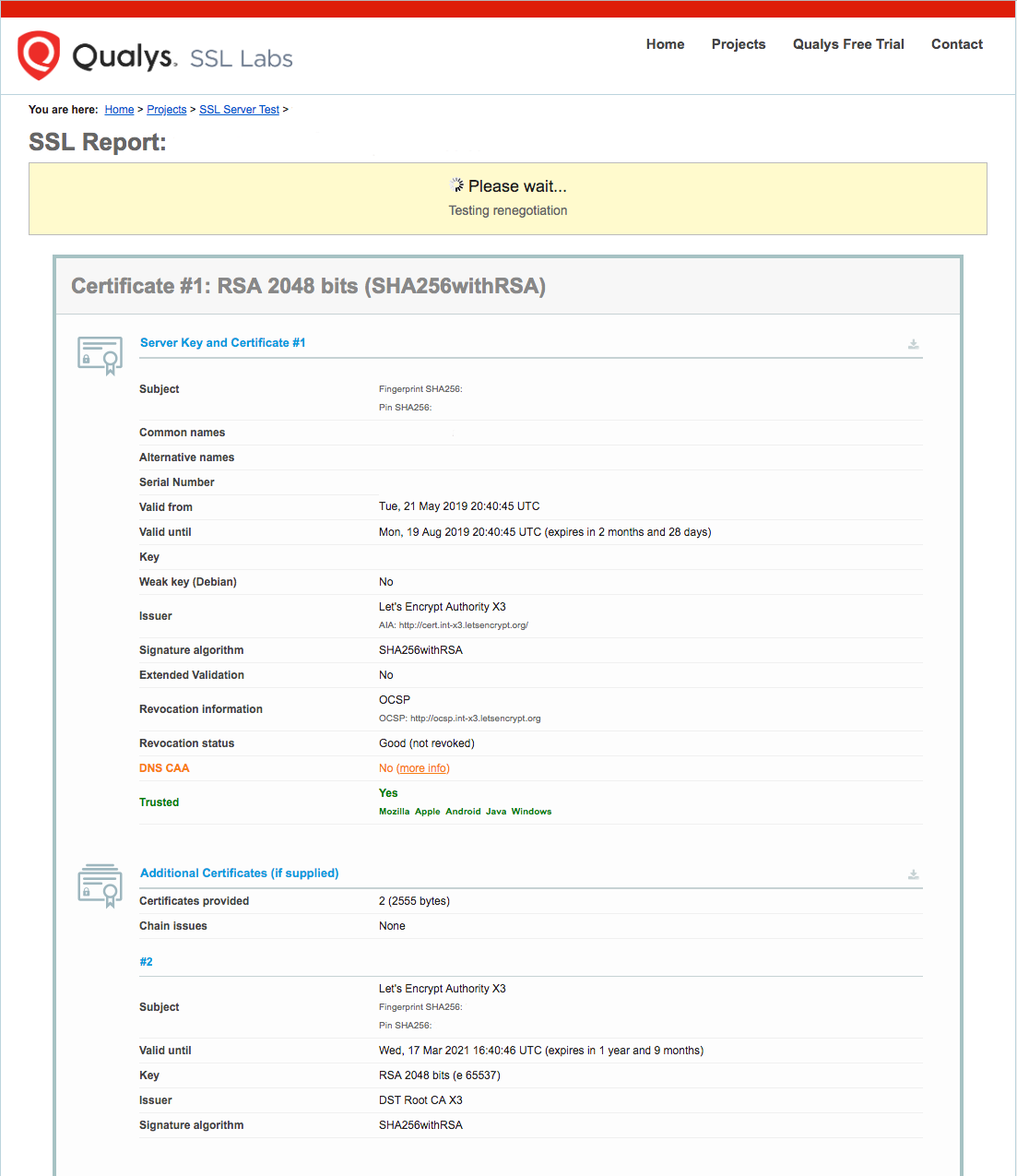
**Step 3 — Checking your Certificate Status**

At this point, you can ensure that Certbot created your SSL certificate correctly by using the [SSL Server Test](https://www.ssllabs.com/ssltest/) from the cloud security company [Qualys](https://www.qualys.com/).

Open the following link in your preferred web browser, replacing example.com with your **base** domain:

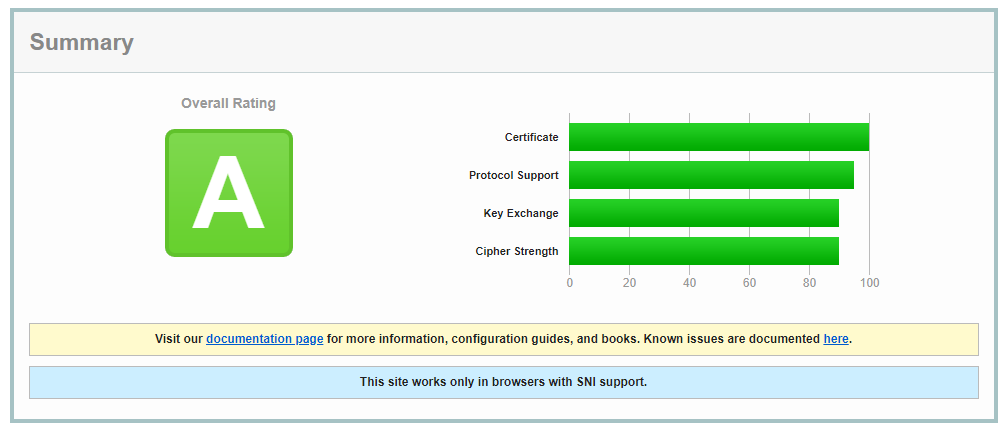
https://www.ssllabs.com/ssltest/analyze.html?d=example.com

You will land on a page that immediately begins testing the SSL connection to your server:



Once the test starts running, it may take a few minutes to complete. The status of the test will update in your browser.

When the testing finishes, the page will display a letter grade that rates the security and quality of your server’s configuration. At the time of this writing, default settings will give an **A** rating:



For more information about how SSL Labs determines these grades, check out the [SSL Labs Grading post](https://community.qualys.com/docs/DOC-6321-ssl-labs-grading-2018) detailing the updates made to the grading scheme in January, 2018.

Try reloading your website using https:// and notice your browser’s security indicator. It will now indicate that the site is properly secured, usually with a green lock icon.

With your SSL certificate up and verified, the next step is to set up auto-renewal for your certificate to keep your certificate valid.

**Step 4 — Setting Up Auto Renewal**

Let’s Encrypt certificates are valid for 90 days, but it’s recommended that you renew the certificates every 60 days to allow a margin of error. Because of this, it is a best practice to automate this process to periodically check and renew the certificate.

First, let’s examine the command that you will use to renew the certificate. The certbot Let’s Encrypt client has a renew command that automatically checks the currently installed certificates and tries to renew them if they are less than 30 days away from the expiration date. By using the --dry-run option, you can run a simulation of this task to test how renew works:

sudo certbot renew --dry-run

The output should look similar to this:

Output

Saving debug log to /var/log/letsencrypt/letsencrypt.log

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Processing /etc/letsencrypt/renewal/example.com.conf

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Cert not due for renewal, but simulating renewal for dry run

Plugins selected: Authenticator apache, Installer apache

Starting new HTTPS connection (1): acme-staging-v02.api.letsencrypt.org

Renewing an existing certificate

Performing the following challenges:

http-01 challenge for example.com

http-01 challenge for www.example.com

Waiting for verification...

Cleaning up challenges

Resetting dropped connection: acme-staging-v02.api.letsencrypt.org

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

new certificate deployed with reload of apache server; fullchain is

/etc/letsencrypt/live/example.com/fullchain.pem

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

\*\* DRY RUN: simulating 'certbot renew' close to cert expiry

\*\* (The test certificates below have not been saved.)

Congratulations, all renewals succeeded. The following certs have been renewed:

/etc/letsencrypt/live/example.com/fullchain.pem (success)

...

Notice that if you created a bundled certificate with multiple domains, only the base domain name will be shown in the output, but the renewal will be valid for all domains included in this certificate.

A practical way to ensure your certificates will not get outdated is to create a [cron job](https://www.digitalocean.com/community/tutorials/how-to-use-cron-to-automate-tasks-on-a-vps) that will periodically execute the automatic renewal command for you. Since the renewal first checks for the expiration date and only executes the renewal if the certificate is less than 30 days away from expiration, it is safe to create a cron job that runs every week or even every day.

The [official Certbot documentation](https://certbot.eff.org/lets-encrypt/centosrhel7-apache) recommends running cron twice per day. This will ensure that, in case Let’s Encrypt initiates a certificate revocation, there will be no more than half a day before Certbot renews your certificate.

Edit the crontab to create a new job that will run the renewal twice per day. To edit the crontab for the **root** user, run:

* sudo crontab -e

Your text editor will open the default crontab which is an empty text file at this point. This tutorial will use the vi text editor. To learn more about this text editor and its successor *vim*, check out our [Installing and Using the Vim Text Editor on a Cloud Server](https://www.digitalocean.com/community/tutorials/installing-and-using-the-vim-text-editor-on-a-cloud-server#managing-documents) tutorial.

Enter insert mode by pressing i and add in the following line:

crontab

0 0,12 \* \* \* python -c 'import random; import time; time.sleep(random.random() \* 3600)' && certbot renew

When you’re finished, press ESC to leave insert mode, then :wq and ENTER to save and exit the file. This will create a new cron job that will execute at noon and midnight every day. Adding an element of randomness to your cron jobs will ensure that hourly jobs do not all happen at the same minute, causing a server spike; python -c 'import random; import time; time.sleep(random.random() \* 3600)' will select a random minute within the hour for your renewal tasks.

For more information on how to create and schedule cron jobs, you can check our [How to Use Cron to Automate Tasks in a VPS](https://www.digitalocean.com/community/tutorials/how-to-use-cron-to-automate-tasks-on-a-vps) guide. More detailed information about renewal can be found in the [Certbot documentation](https://certbot.eff.org/docs/using.html" \l "renewal).

Conclusion

In this guide you installed the Let’s Encrypt Certbot client, downloaded SSL certificates for your domain, and set up automatic certificate renewal. If you have any questions about using Certbot, you can check the official [Certbot documentation](https://certbot.eff.org/docs/). We also recommend that you check the official [Let’s Encrypt blog](https://letsencrypt.org/blog/) for important updates from time to time.

# How To Install and Configure Zabbix to Securely Monitor Remote Servers on CentOS 7

Step 1 — Installing the Zabbix Server

First, you need to install Zabbix on the server where you installed MySQL, Apache, and PHP. Log in to this machine as your non-root user:

Zabbix isn’t available in the package manager by default, so install a repository configuration package using the [official Zabbix repository](https://repo.zabbix.com/) for CentOS. This tutorial will use version 4.2 of Zabbix:

sudo rpm -Uvh https://repo.zabbix.com/zabbix/4.2/rhel/7/x86\_64/zabbix-release-4.2-1.el7.noarch.rpm

You will see the following output:

Output

Retrieving https://repo.zabbix.com/zabbix/4.2/rhel/7/x86\_64/zabbix-release-4.2-1.el7.noarch.rpm

warning: /var/tmp/rpm-tmp.WXsYNB: Header V4 RSA/SHA512 Signature, key ID a14fe591: NOKEY

Preparing... ################################# [100%]

Updating / installing...

1:zabbix-release-4.2-1.el7 ################################# [100%]

Clear all entries for currently enabled repositories from the cache:

sudo yum clean all

Then install the Zabbix server and web frontend with MySQL database support:

sudo yum install zabbix-server-mysql zabbix-web-mysql

During the installation process you will be asked about importing a GPG key. This key will verify the authenticity of the package you are installing. To allow the installation to finish, accept the GPG key by typing y and pressing ENTER when prompted to do so.

Also, install the Zabbix agent, which will let you collect data about the Zabbix server status itself.

sudo yum install zabbix-agent

Before you can use Zabbix, you have to set up a database to hold the data that the Zabbix server will collect from its agents. You can do this in the next step.

**Step 2 — Configuring the MySQL Database for Zabbix**

You need to create a new MySQL database and populate it with some basic information in order to make it suitable for Zabbix. You’ll also create a specific user for this database so Zabbix isn’t logging in to MySQL with the root account.

Log in to MySQL as the **root** user using the **root** password that you set up during the MySQL server installation:

mysql -uroot -p

Create the Zabbix database with UTF-8 character support:

create database zabbix character set utf8 collate utf8\_bin;

Then create a user that the Zabbix server will use, give it access to the new database, and set the password for the user:

grant all privileges on zabbix.\* to zabbix@localhost identified by 'Training123';

Then apply these new permissions:

flush privileges;

That takes care of the user and the database. Exit out of the database console:

quit;

Next, import the initial schema and data. The Zabbix installation provided you with a file that sets this up.

Run the following command to set up the schema and import the data into the zabbix database. Use zcat since the data in the file is compressed.

zcat /usr/share/doc/zabbix-server-mysql\*/create.sql.gz | mysql -uzabbix -p zabbix

Enter the password for the zabbix MySQL user that you configured when prompted.

This command will not output any errors if it was successful. If you see the error ERROR 1045 (28000): Access denied for user zabbix@'localhost' (using password: YES) then make sure you used the password for the **zabbix** user and not the **root** user.

In order for the Zabbix server to use this database, you need to set the database password in the Zabbix server configuration file. Open the configuration file in your preferred text editor. This tutorial will use vi:

sudo vi /etc/zabbix/zabbix\_server.conf

Look for the following section of the file:

/etc/zabbix/zabbix\_server.conf

...

### Option: DBPassword

# Database password. Ignored for SQLite.

# Comment this line if no password is used.

#

# Mandatory: no

# Default:

# DBPassword=

...

These comments in the file explain how to connect to the database. You need to set the DBPassword value in the file to the password for your database user. Enter insert mode by pressing i, uncomment DBPassword=, and add your password to configure the database:

/etc/zabbix/zabbix\_server.conf

...

# Mandatory: no

# Default

DBPassword=your\_zabbix\_mysql\_password

...

When you’re finished, press ESC to leave insert mode, then :wq and ENTER to save and exit the file.

That takes care of the Zabbix server configuration. Next, you will make some modifications to your PHP setup in order for the Zabbix web interface to work properly.

**Step 3 — Configuring PHP for Zabbix**

The Zabbix web interface is written in PHP and requires some special PHP server settings. The Zabbix installation process created an Apache configuration file that contains these settings. You need to make a small change to this file, so open it up with the following:

sudo vi /etc/httpd/conf.d/zabbix.conf

The file contains PHP settings that meet the necessary requirements for the Zabbix web interface. However, the timezone setting is commented out by default. To make sure that Zabbix uses the correct time, you need to set the appropriate timezone.

/etc/httpd/conf.d/zabbix.conf

...

<IfModule mod\_php5.c>

php\_value max\_execution\_time 300

php\_value memory\_limit 128M

php\_value post\_max\_size 16M

php\_value upload\_max\_filesize 2M

php\_value max\_input\_time 300

php\_value max\_input\_vars 10000

php\_value always\_populate\_raw\_post\_data -1

# php\_value date.timezone Europe/Riga

</IfModule>

Uncomment the timezone line, highlighted in the preceding code block, and change it to your timezone. You can use this [list of supported time zones](http://php.net/manual/en/timezones.php) to find the right one for you. Then save and close the file. You can use this to confirm your timezone $ timedatectl

Now restart Apache to apply these new settings:

sudo systemctl restart httpd

**Note:** If SELinux is running in enforcing mode, you need to put it into permissive mode using the sudo setenforce 0 command to allow the Zabbix agent to operate freely. You can use [this tutorial](https://www.digitalocean.com/community/tutorials/an-introduction-to-selinux-on-centos-7-part-1-basic-concepts) to get more info about the Security Enhanced Linux control mechanism.

You can now start the Zabbix server and agent:

sudo systemctl start zabbix-server

sudo systemctl start zabbix-agent

Then check whether the Zabbix server is running properly:

sudo systemctl status zabbix-server

You will see the following status:

Output

● zabbix-server.service - Zabbix Server

Loaded: loaded (/usr/lib/systemd/system/zabbix-server.service; disabled; vendor preset: disabled)

Active: active (running) since Fri 2019-05-03 05:57:29 UTC; 2s ago

Process: 4461 ExecStart=/usr/sbin/zabbix\_server -c $CONFFILE (code=exited, status=0/SUCCESS)

...

Finally, enable the server and agent to start at boot time:

sudo systemctl enable zabbix-server

sudo systemctl enable zabbix-agent

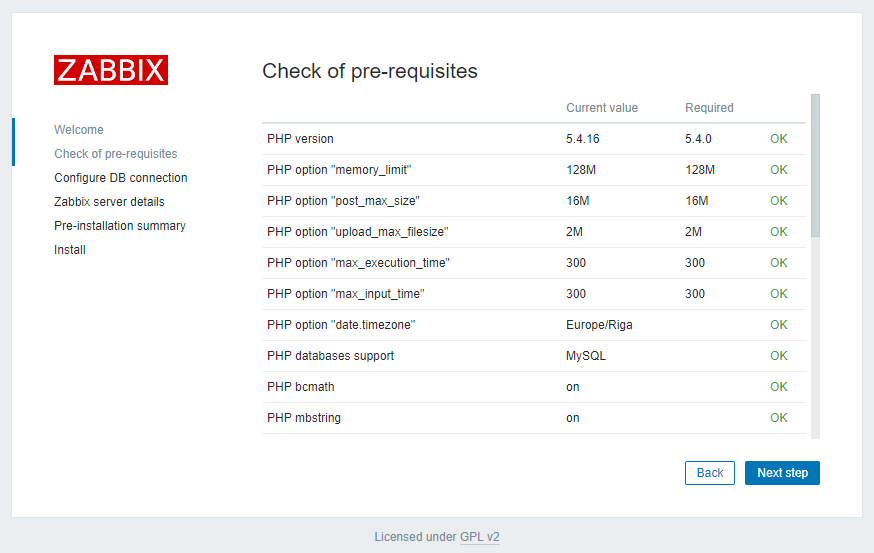
The server is set up and connected to the database. Next, set up the web frontend.

**Note:** As mentioned in the Prerequisites section, it is recommended that you enable SSL/TLS on your server. You can follow [this tutorial](https://www.digitalocean.com/community/tutorials/how-to-secure-apache-with-let-s-encrypt-on-centos-7) now to obtain a free SSL certificate for Apache on CentOS 7. After obtaining your SSL/TLS certificates, you can come back and complete this tutorial.

**Step 4 — Configuring Settings for the Zabbix Web Interface**

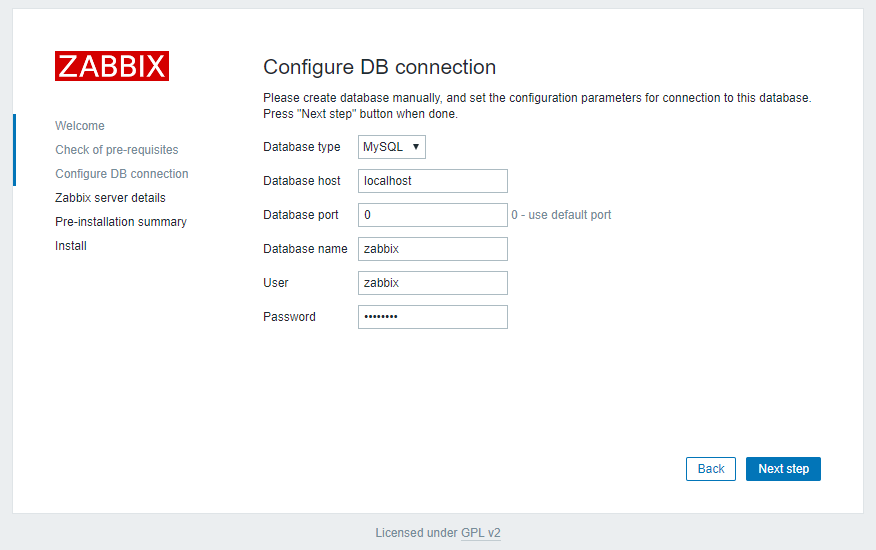
The web interface lets you see reports and add hosts that you want to monitor, but it needs some initial setup before you can use it. Launch your browser and go to the address **http://zabbix\_server\_name/zabbix/.** On the first screen, you will see a welcome message. Click **Next step** to continue.

On the next screen, you will see the table that lists all of the prerequisites to run Zabbix.



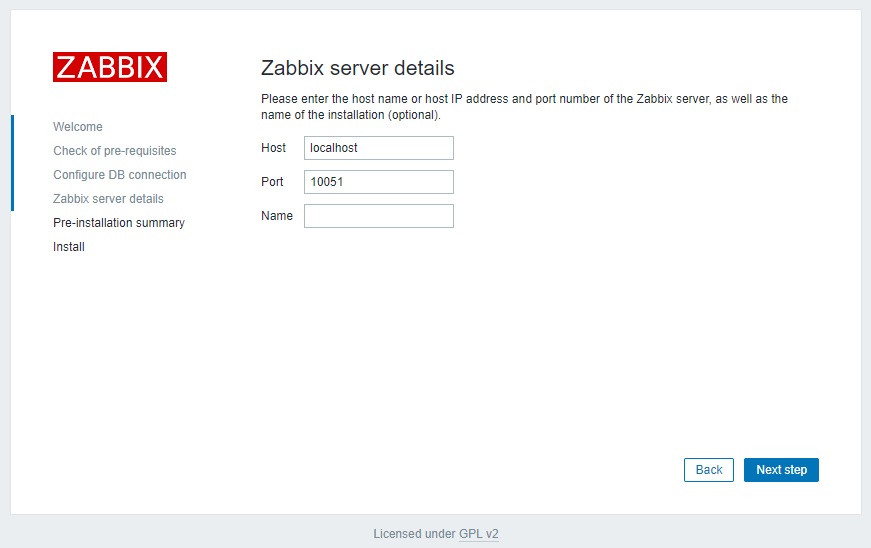
All of the values in this table must be **OK**, so verify that they are. Be sure to scroll down and look at all of the prerequisites. Once you’ve verified that everything is ready to go, click **Next step** to proceed.

The next screen asks for database connection information.



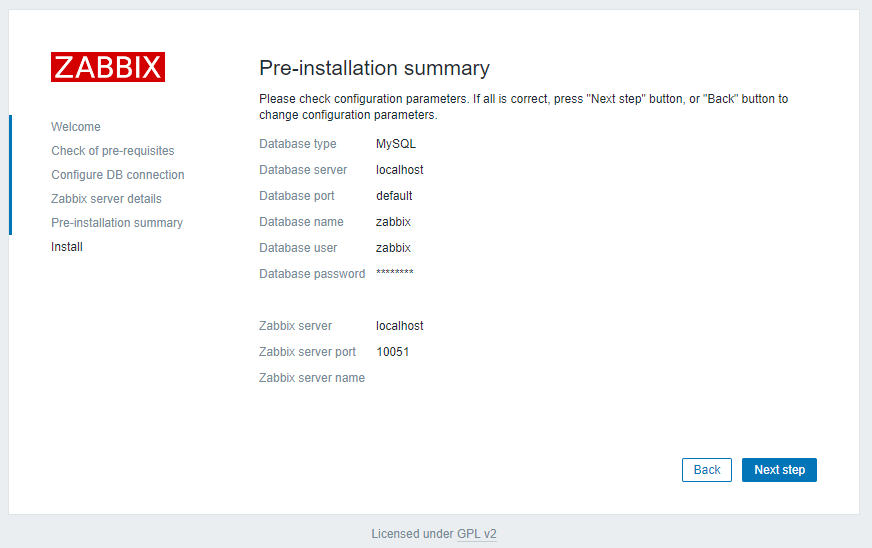
You told the Zabbix server about your database, but the Zabbix web interface also needs access to the database to manage hosts and read data. Therefore enter the MySQL credentials you configured in Step 2 and click **Next step** to proceed.

On the next screen, you can leave the options at their default values.



The **Name** is optional; it is used in the web interface to distinguish one server from another in case you have several monitoring servers. Click **Next step** to proceed.

The next screen will show the pre-installation summary so you can confirm everything is correct.



Click **Next step** to proceed to the final screen.

The web interface setup is complete. This process creates the configuration file /etc/zabbix/web/zabbix.conf.php which you could back up and use in the future. Click **Finish** to proceed to the login screen. The default user is **Admin** and the password is **zabbix**.

Before you log in, set up the Zabbix agent on your second CentOS server.

**Step 5 — Installing and Configuring the Zabbix Agent**

Now you need to configure the agent software that will send monitoring data to the Zabbix server.

**Log in to the second CentOS server – Zabbix Agent:**

* ssh sammy@second\_centos\_server\_ip\_address

Then, just like on the Zabbix server, run the following command to install the repository configuration package:

sudo rpm -Uvh https://repo.zabbix.com/zabbix/4.2/rhel/7/x86\_64/zabbix-release-4.2-1.el7.noarch.rpm

Next, clear the yum cache:

sudo yum clean all

Then install the Zabbix agent:

sudo yum -y install zabbix-agent

During the installation process you will be asked about importing a GPG key. Confirm it so the installation can complete.

While Zabbix supports certificate-based encryption, setting up a certificate authority is beyond the scope of this tutorial. Instead, this tutorial will use [*pre-shared keys*](https://www.zabbix.com/documentation/4.2/manual/encryption/using_pre_shared_keys) (PSK) to secure the connection between the server and agent.

First, generate a PSK:

sudo sh -c "openssl rand -hex 32 > /etc/zabbix/zabbix\_agentd.psk"

Show the key so you can copy it somewhere. You will need it to configure the host.

cat /etc/zabbix/zabbix\_agentd.psk

The key will look something like this:

Output

a4374c8c3b55de472225dbf96aa7cd3ce6a34abc95d0eb2a601d954ee0cd4410

Now edit the Zabbix agent settings to set up its secure connection to the Zabbix server. Open the agent configuration file in your text editor:

sudo vi /etc/zabbix/zabbix\_agentd.conf

Each setting within this file is documented via informative comments throughout the file, but you only need to edit some of them.

First you have to edit the IP address of the Zabbix server. Find the following section:

/etc/zabbix/zabbix\_agentd.conf

...

### Option: Server

# List of comma delimited IP addresses (or hostnames) of Zabbix servers.

# Incoming connections will be accepted only from the hosts listed here.

# If IPv6 support is enabled then '127.0.0.1', '::127.0.0.1', '::ffff:127.0.0.1' are treated equally.

#

# Mandatory: no

# Default:

# Server=

Server=127.0.0.1

...

Change the default value to the IP of your Zabbix server:

/etc/zabbix/zabbix\_agentd.conf

...

# Mandatory: no

# Default:

# Server=

Server=zabbix\_server\_ip\_address

...

Next, find the section that configures the secure connection to the Zabbix server and enable pre-shared key support. Find the TLSConnect section, which looks like this:

/etc/zabbix/zabbix\_agentd.conf

...

### Option: TLSConnect

# How the agent should connect to server or proxy. Used for active checks.

# Only one value can be specified:

# unencrypted - connect without encryption

# psk - connect using TLS and a pre-shared key

# cert - connect using TLS and a certificate

#

# Mandatory: yes, if TLS certificate or PSK parameters are defined (even for 'unencrypted' connection)

# Default:

# TLSConnect=unencrypted

...

Then uncomment TLSConnect= and replace unencrypted with psk to configure pre-shared key support:

/etc/zabbix/zabbix\_agentd.conf

...

TLSConnect=psk

...

Next, locate the TLSAccept section, which looks like this:

/etc/zabbix/zabbix\_agentd.conf

...

### Option: TLSAccept

# What incoming connections to accept.

# Multiple values can be specified, separated by comma:

# unencrypted - accept connections without encryption

# psk - accept connections secured with TLS and a pre-shared key

# cert - accept connections secured with TLS and a certificate

#

# Mandatory: yes, if TLS certificate or PSK parameters are defined (even for 'unencrypted' connection)

# Default:

# TLSAccept=unencrypted

...

Configure incoming connections to support pre-shared keys by uncommenting TLSAccept= and adding psk:

/etc/zabbix/zabbix\_agentd.conf

...

TLSAccept=psk

...

Next, find the TLSPSKIdentity section, which looks like this:

/etc/zabbix/zabbix\_agentd.conf

...

### Option: TLSPSKIdentity

# Unique, case sensitive string used to identify the pre-shared key.

#

# Mandatory: no

# Default:

# TLSPSKIdentity=

...

Choose a unique name to identify your pre-shared key by uncommenting TLSPSKIdentity= and adding the highlighted code:

/etc/zabbix/zabbix\_agentd.conf

...

TLSPSKIdentity=PSK 001

...

You’ll use this as the **PSK ID** when you add your host through the Zabbix web interface.

Then set the option that points to your previously created pre-shared key. Locate the TLSPSKFile option:

/etc/zabbix/zabbix\_agentd.conf

...

### Option: TLSPSKFile

# Full pathname of a file containing the pre-shared key.

#

# Mandatory: no

# Default:

# TLSPSKFile=

...

Uncomment TLSPSKFile= and add this text to point the Zabbix agent to the PSK file you created:

/etc/zabbix/zabbix\_agentd.conf

...

TLSPSKFile=/etc/zabbix/zabbix\_agentd.psk

...

Save and close the file. Now you can start the Zabbix agent and set it to start at boot time:

sudo systemctl start zabbix-agent

sudo systemctl enable zabbix-agent

For good measure, check that the Zabbix agent is running properly:

sudo systemctl status zabbix-agent

You will see the following status, indicating the agent is running:

Output

● zabbix-agent.service - Zabbix Agent

Loaded: loaded (/usr/lib/systemd/system/zabbix-agent.service; enabled; vendor preset: disabled)

Active: active (running) since Fri 2019-05-03 06:25:06 UTC; 28s ago

...

The agent will listen on port 10050 for connections from the server. In order to configure access coming from specific IP addresses or subnets, use the [rich rule](https://firewalld.org/documentation/man-pages/firewalld.richlanguage.html) functionality of firewalld:

sudo firewall-cmd --permanent --zone=public --add-rich-rule='rule family="ipv4" source address="**zabbix\_server\_ip\_address/32**" port protocol="tcp" port="10050" accept'

Rich rules allow you to create more complex and customizable firewalld rules to gain greater control over your firewall. In this command, you are adding a rule that accepts ipv4 traffic from the source, which you have set as the IP address of the Zabbix server, to port 10050 of your second CentOS server.

Next, reload firewalld to activate the new rule:

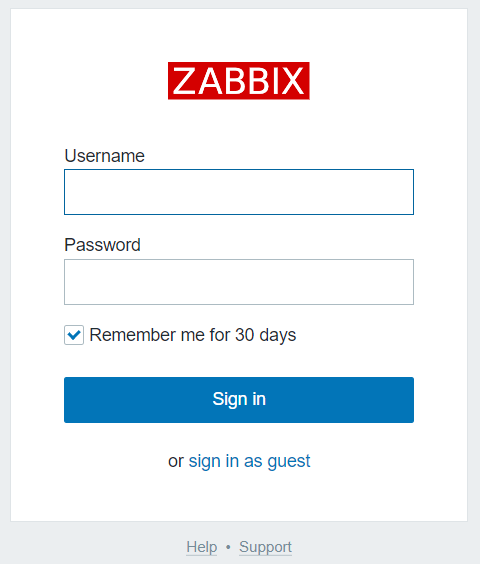
sudo firewall-cmd --reload

Your agent is now ready to accept connections and send data to the Zabbix server. But in order to use it, you have to link to it from the server’s web console. In the next step, you will complete the configuration.

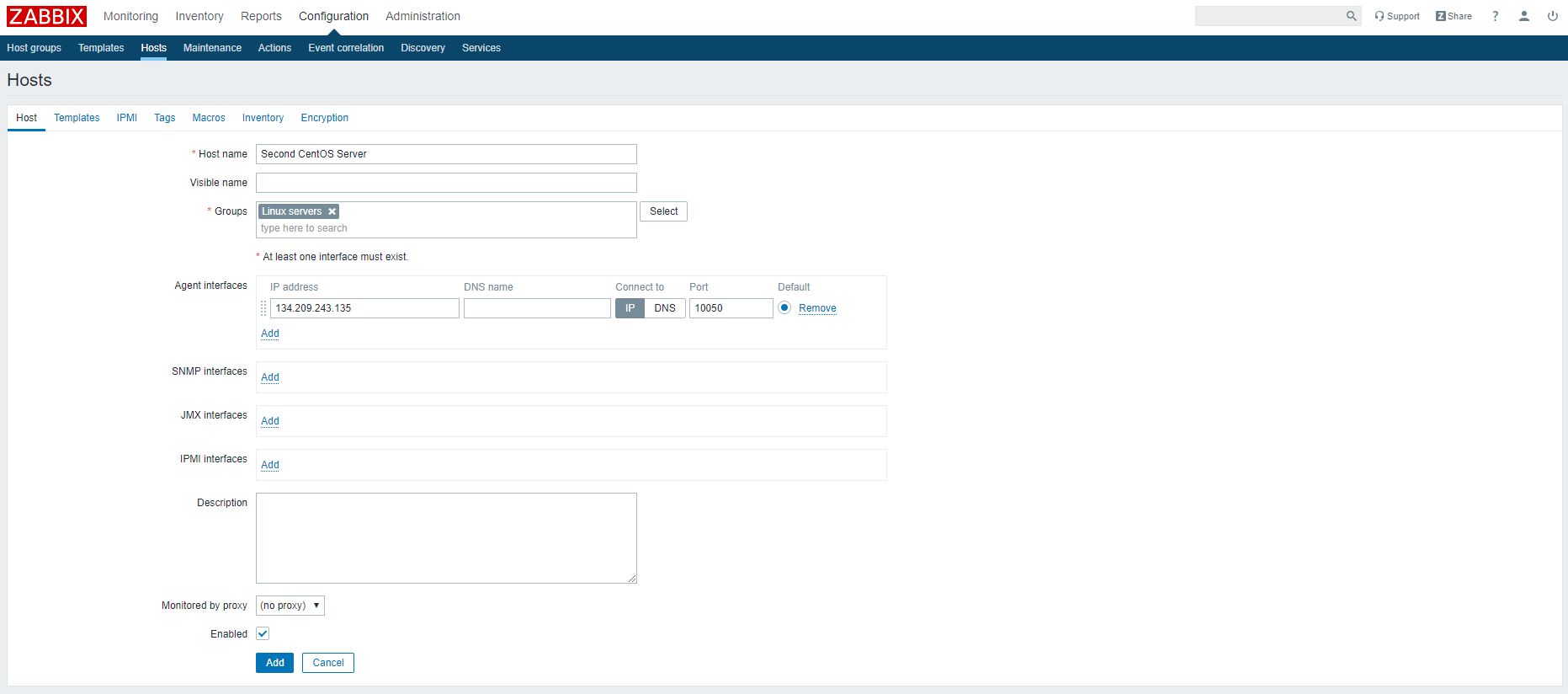
**Step 6 — Adding the New Host to the Zabbix Server**

Installing an agent on a server you want to monitor is only half of the process. Each host you want to monitor needs to be registered on the Zabbix server, which you can do through the web interface.

Log in to the Zabbix server web interface by navigating to the address http://zabbix\_server\_name/zabbix/.

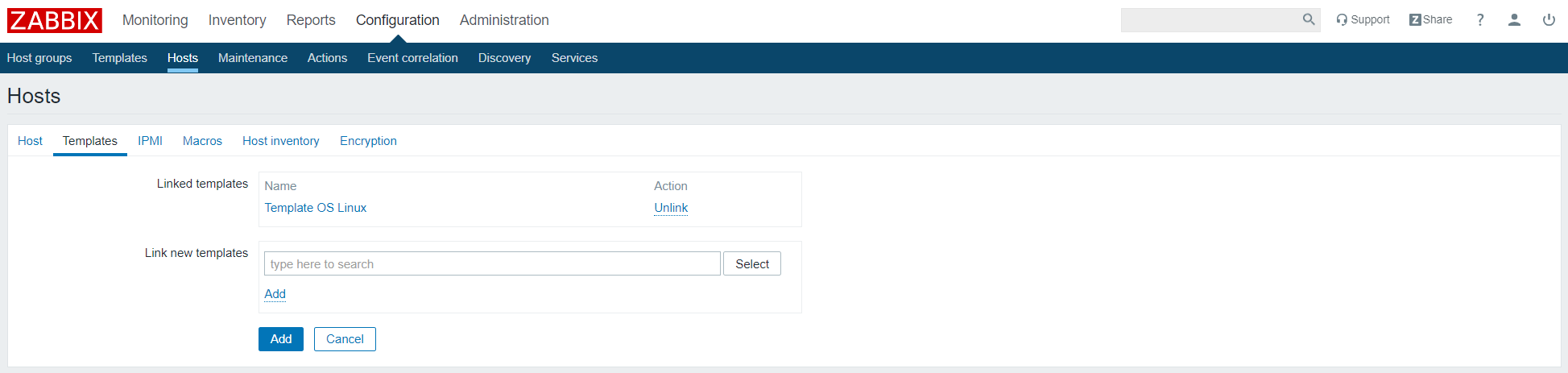


When you have logged in, click on **Configuration**, and then **Hosts** in the top navigation bar. Then click the **Create host** button in the top right corner of the screen. This will open the host configuration page.



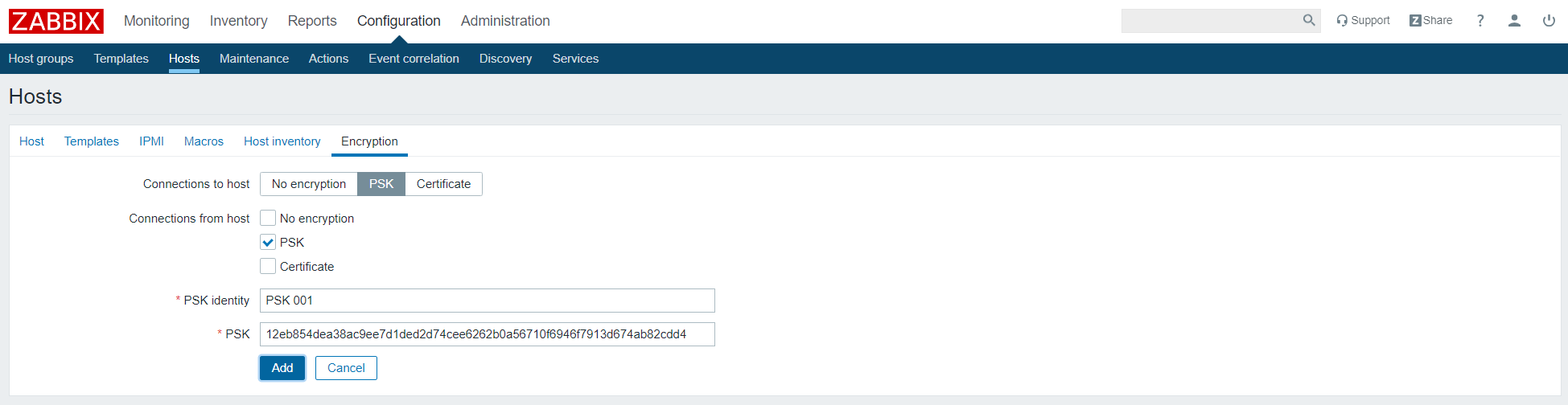
Adjust the **Host name** and **IP address** to reflect the host name and IP address of your second CentOS server, then add the host to a group. You can select an existing group, for example **Linux servers**, or create your own group. The host can be in multiple groups. To do this, enter the name of an existing or new group in the **Groups** field and select the desired value from the proposed list.

Once you’ve added the group, click the **Templates** tab.



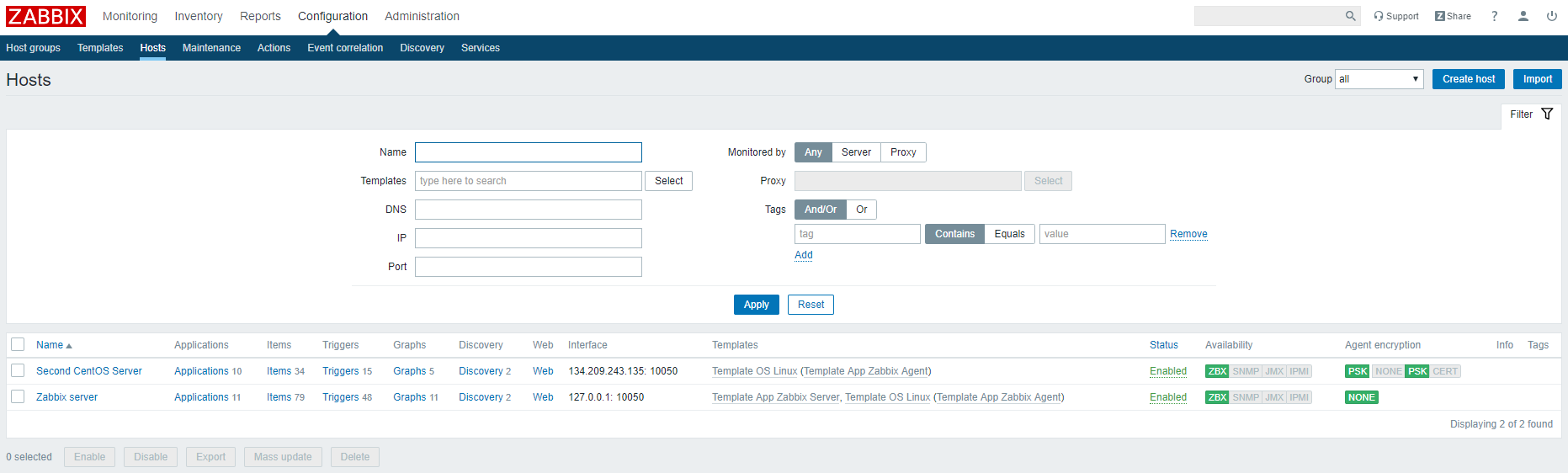
Type Template OS Linux in the **Search** field and then click **Add** immediately under the search bar to add this template to the host.

Next, navigate to the **Encryption** tab. Select **PSK** for both **Connections to host** and **Connections from host** and uncheck **No encryption** for **Connections from host**. Then set **PSK identity** to PSK 001, which is the value of the **TLSPSKIdentity** setting of the Zabbix agent you configured previously. Then set **PSK** value to the key you generated for the Zabbix agent. It’s the one stored in the file /etc/zabbix/zabbix\_agentd.psk on the agent machine.



Finally, click the **Add** button at the bottom of the form to create the host.

You will see your new host in the list. Wait for a minute and reload the page to see green labels indicating that everything is working fine and the connection is encrypted.



If you have additional servers you need to monitor, log in to each host, install the Zabbix agent, generate a PSK, configure the agent, and add the host to the web interface following the same steps you followed to add your first host.

The Zabbix server is now monitoring your second CentOS server. Now, set up email notifications to be notified about problems.

**Step 7 — Configuring Email Notifications**

Zabbix automatically supports several types of notifications: email, [Jabber](https://www.jabber.org/), SMS, etc. You can also use alternative notification methods, such as Telegram or Slack. You can see the full list of integrations [here](https://www.zabbix.com/ru/integrations/?cat=notifications_alerting).

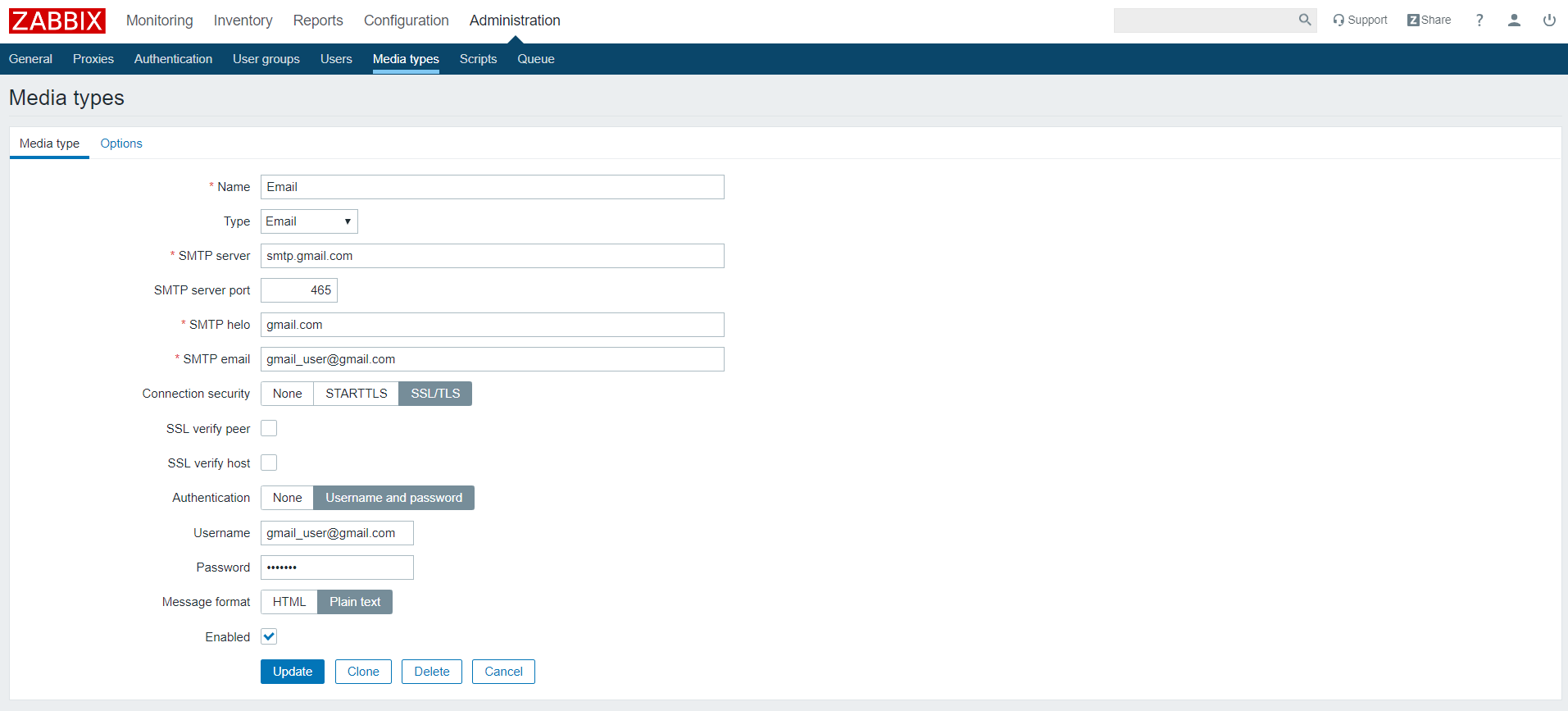
The simplest communication method is email, and this tutorial will configure notifications for this media type.

Click on **Administration**, and then **Media types** in the top navigation bar. You will see the list of all media types. Click on **Email**.

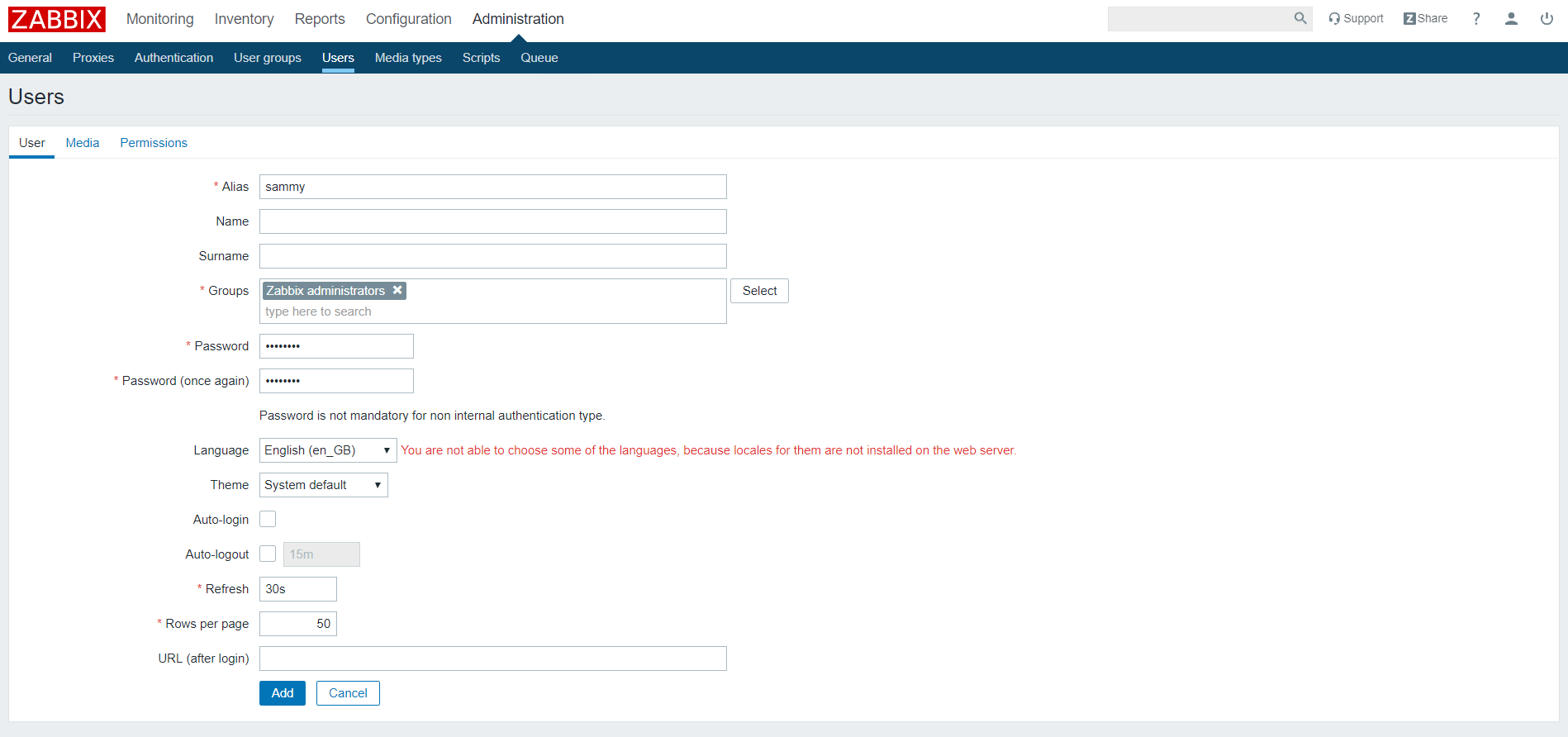
Adjust the SMTP options according to the settings provided by your email service. This tutorial uses Gmail’s SMTP capabilities to set up email notifications; if you would like more information about setting this up, see [How To Use Google’s SMTP Server](https://www.digitalocean.com/community/tutorials/how-to-use-google-s-smtp-server).

**Note:** If you use 2-Step Verification with Gmail, you need to generate an App Password for Zabbix. You don’t need to remember it, you’ll only have to enter an App password once during setup. You will find instructions on how to generate this password in the [Google Help Center](https://support.google.com/accounts/answer/185833?hl=en).

You can also choose the message format—html or plain text. Finally, click the **Update** button at the bottom of the form to update the email parameters.

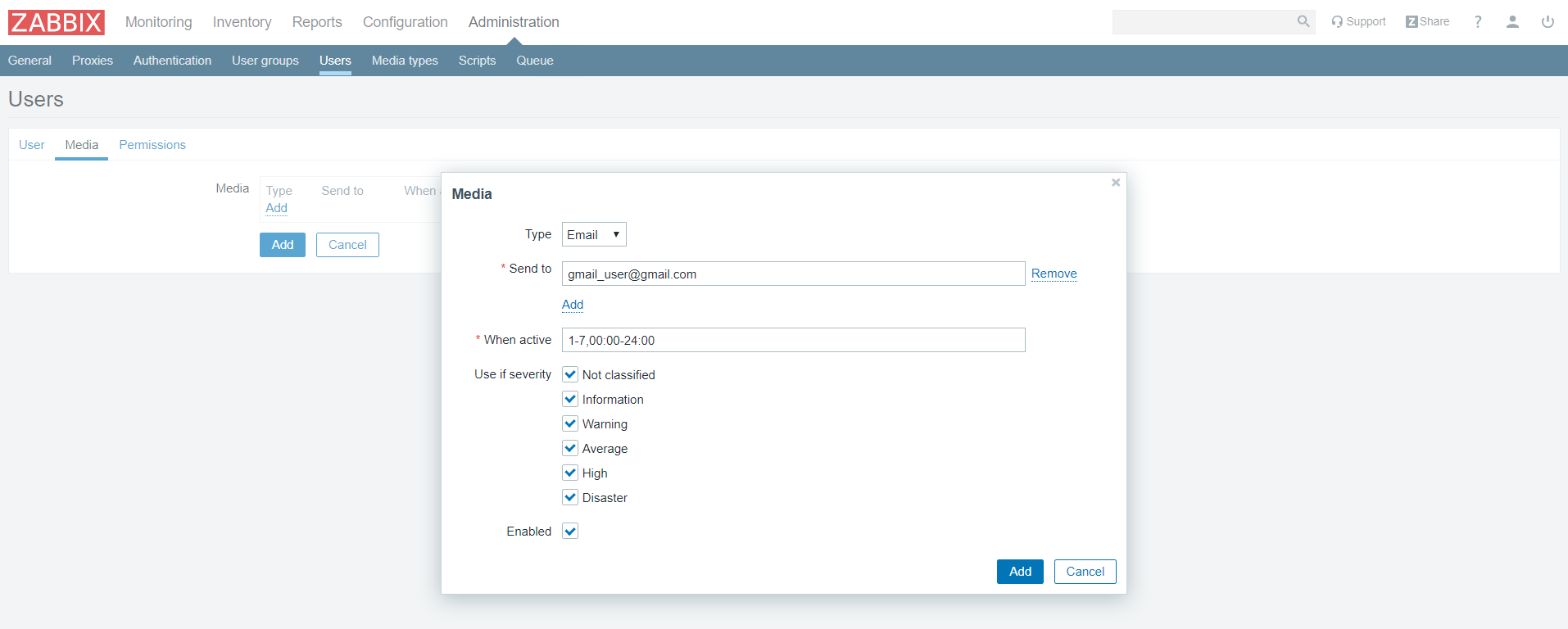


Now, create a new user. Click on **Administration**, and then **Users** in the top navigation bar. You will see the list of users. Then click the **Create user** button in the top right corner of the screen. This will open the user configuration page.



Enter the new username in the **Alias** field and set up a new password. Next, add the user to the administrator’s group. Type Zabbix administrators in the **Groups** field and select it from the proposed list.

Once you’ve added the group, click the **Media** tab and click on the **Add** underlined link. You will see a pop-up window.



Enter your email address in the **Send to** field. You can leave the rest of the options at the default values. Click the **Add** button at the bottom to submit.

Now navigate to the **Permissions** tab. Select **Zabbix Super Admin** from the **User type** drop-down menu.

Finally, click the **Add** button at the bottom of the form to create the user.

Now you need to enable notifications. Click on the **Configuration** tab, and then **Actions** in the top navigation bar. You will see a pre-configured action, which is responsible for sending notifications to all Zabbix administrators. You can review and change the settings by clicking on its name. For the purposes of this tutorial, use the default parameters. To enable the action, click on the red **Disabled** link in the **Status** column.

Now you are ready to receive alerts. In the next step, you will generate one to test your notification setup.

**Step 8 — Generating a Test Alert**

In this step, you will generate a test alert to ensure everything is connected. By default, Zabbix keeps track of the amount of free disk space on your server. It automatically detects all disk mounts and adds the corresponding checks. This discovery is executed every hour, so you need to wait a while for the notification to be triggered.

Create a temporary file that’s large enough to trigger Zabbix’s file system usage alert. To do this, log in to your second CentOS server if you’re not already connected.

* ssh sammy@second\_centos\_server\_ip\_address

Next, determine how much free space you have on the server. You can use the df command to find out:

df -h

The command df will report the disk space usage of your file system, and the -h will make the output human-readable. You’ll see output like the following:

Output

Filesystem Size Used Avail Use% Mounted on

/dev/vda1 25G 958M 25G 4% /

In this case, the free space is 25GB. Your free space may differ.

Use the fallocate command, which allows you to pre-allocate or de-allocate space to a file, to create a file that takes up more than 80% of the available disk space. This will be enough to trigger the alert:

fallocate -l 20G /tmp/temp.img

After around an hour, Zabbix will trigger an alert about the amount of free disk space and will run the action you configured, sending the notification message. You can check your inbox for the message from the Zabbix server. You will see a message like:

Problem started at 10:49:25 on 2019.05.03

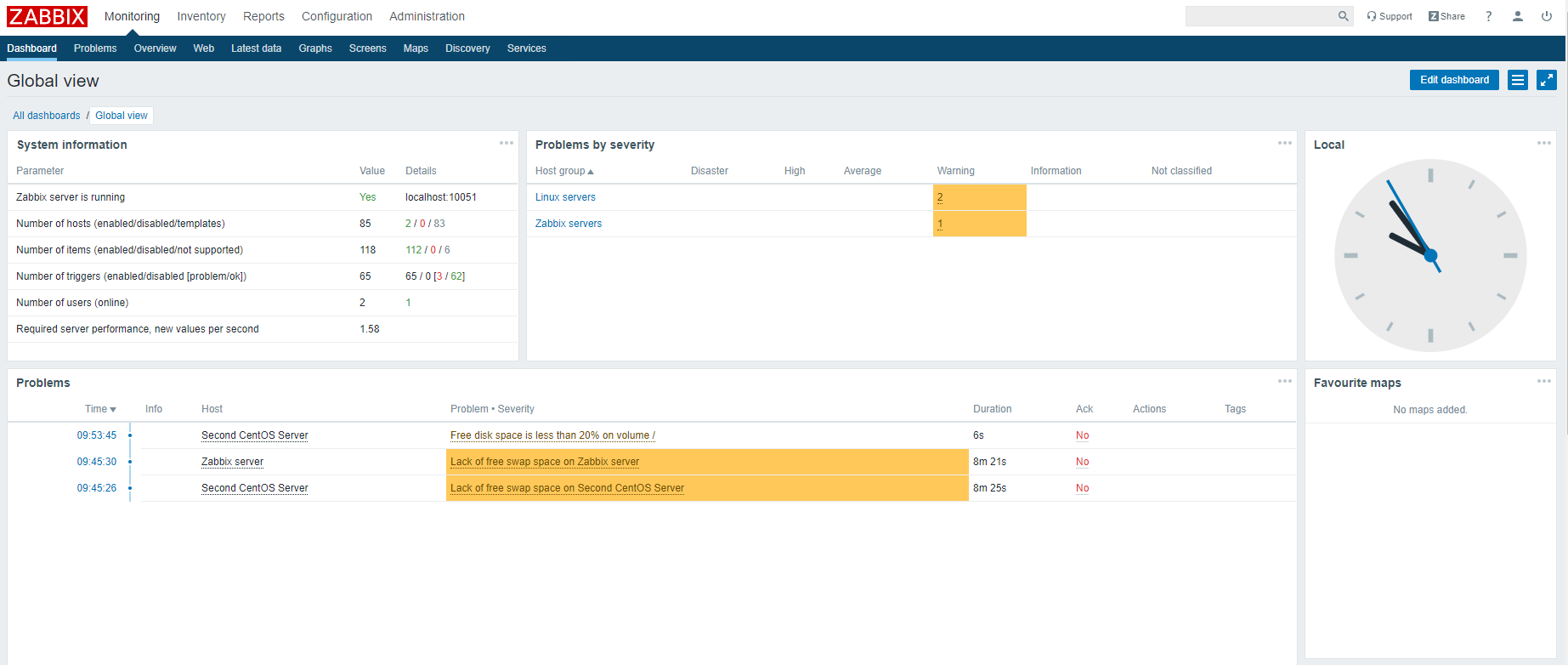
Problem name: Free disk space is less than 20% on volume /

Host: Second Centos Server

Severity: Warning

Original problem ID: 34

You can also navigate to the **Monitoring** tab, and then **Dashboard** to see the notification and its details.



Now that you know the alerts are working, delete the temporary file you created so you can reclaim your disk space:

rm -f /tmp/temp.img

After a minute Zabbix will send the recovery message and the alert will disappear from the main dashboard.

**Conclusion**

In this tutorial, you set up a simple and secure monitoring solution that will help you monitor the state of your servers. It can now warn you of problems, and you have the opportunity to analyze the processes occurring in your IT infrastructure.

To learn more about setting up monitoring infrastructure, check out [How To Install Elasticsearch, Logstash, and Kibana (Elastic Stack) on CentOS 7](https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elastic-stack-on-centos-7) and [How To Gather Infrastructure Metrics with Metricbeat on CentOS 7](https://www.digitalocean.com/community/tutorials/how-to-gather-infrastructure-metrics-with-metricbeat-on-centos-7).