

# How to Install and Secure Grafana on Ubuntu 16.04

Posted December 27, 2017  43k

DATA ANALYSIS

UBUNTU 16.04



By: Marko Mudrinić

## Introduction

Grafana is an open-source, data visualization and monitoring tool that integrates with complex data from sources like Prometheus, InfluxDB, Graphite, and ElasticSearch. Grafana lets you create alerts, notifications, and ad-hoc filters for your data while also making collaboration with your teammates easier through built-in sharing features.

In this tutorial, you will install Grafana and secure it with an SSL certificate and an Nginx reverse proxy, then you'll modify Grafana's default settings for even tighter security.

## Prerequisites

To follow this tutorial, you will need:

- One Ubuntu 16.04 server set up by following the Initial Server Setup with Ubuntu 16.04 tutorial, including a `sudo` non-root user and a firewall.
- A fully registered domain name. This tutorial uses `example.com` throughout. You can purchase a domain name on Namecheap, get one for free on Freenom, or use the domain registrar of your choice.
- The following DNS records set up for your server. You can follow How To Set Up a Host Name with DigitalOcean for details on how to add them.
  - An **A** record with `example.com` pointing to your server's public IP address.
  - An **A** record with `www.example.com` pointing to your server's public IP address.
- Nginx set up by following the first two steps of the How To Install Nginx on Ubuntu 16.04 tutorial.
- An Nginx Server Block with Let's Encrypt configured, which can be set up by following How To Set Up Let's Encrypt with Nginx Server Blocks on Ubuntu 16.04.
- Optionally, to set up GitHub authentication, you'll need a GitHub account associated with an organization.

# Step 1 — Installing Grafana

You can install Grafana either by downloading it directly from its official website or by going through an APT repository. Because an APT repository makes it easier to install and manage Grafana's updates, we'll use that method.

Although Grafana is available in the official Ubuntu 16.04 packages repository, the version of Grafana there may not be the latest, so we'll use Grafana's official repository on packagecloud.

Download the packagecloud GPG key with `curl`, then pipe the output to apt-key. This will add the key to your APT installation's list of trusted keys, which will allow you to download and verify the GPG-signed Grafana package.

```
$ curl https://packagecloud.io/gpg.key | sudo apt-key add -
```

Next, add the packagecloud repository to your APT sources.

```
$ sudo add-apt-repository "deb https://packagecloud.io/grafana/stable/debian/ stretch main"
```

**Note:** Although this tutorial is written for Ubuntu 16.04, packagecloud only provides Debian, Python, RPM, and RubyGem packages. You can use the Debian-based repository in the previous command, though, because the Grafana package it contains is the same as the one for Ubuntu. Just be sure to use the **stretch** repository to get the latest version of Grafana.

Refresh your APT cache to update your package lists.

```
$ sudo apt-get update
```

And, make sure Grafana will be installed from the packagecloud repository.

```
$ apt-cache policy grafana
```

The output tells you the version of Grafana that will be installed and where the package will be retrieved from. Verify that the installation candidate will come from the official Grafana repository at `https://packagecloud.io/grafana/stable/debian`.

Output of `apt-cache policy grafana`

```
grafana:
  Installed: (none)
  Candidate: 4.6.2
  Version table:
    4.6.2 500
```

...

You can now proceed with the installation.

```
$ sudo apt-get install grafana
```

Once Grafana's installed, you're ready to start it.

```
$ sudo systemctl start grafana-server
```

Next, verify that Grafana is running by checking the service's status.

```
$ sudo systemctl status grafana-server
```

The output contains information about Grafana's process, including its status, Main Process Identifier (PID), memory use, and more.

If the service status isn't **active (running)**, review the output and re-trace the preceding steps to resolve the problem.

Output of `grafana-server status`

- `grafana-server.service` - Grafana instance

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

Active: **active (running)** since Thu 2017-12-07 12:10:33 UTC; 19s ago

Docs: <http://docs.grafana.org>

Main PID: 14796 (grafana-server)

Tasks: 6

Memory: 32.0M

CPU: 472ms

CGroup: /system.slice/grafana-server.service

└─14796 /usr/sbin/grafana-server --config=/etc/grafana/grafana.ini --pidfile=/var/run/graf

...

Lastly, enable the service to automatically start Grafana on boot.

```
$ sudo systemctl enable grafana-server
```

The output confirms that `systemd` has created the necessary symbolic links to autostart Grafana. If you receive an error message, follow the instructions in the terminal to fix the problem before continuing.

Output of `systemctl enable grafana-server`

```
Synchronizing state of grafana-server.service with SysV init with /lib/systemd/systemd-sysv-install.  
Executing /lib/systemd/systemd-sysv-install enable grafana-server  
Created symlink from /etc/systemd/system/multi-user.target.wants/grafana-server.service to /usr/lib/s
```

Grafana is now installed and ready to be used. Next, secure your connection to Grafana with a reverse proxy and SSL certificate.

## Step 2 — Setting Up the Reverse Proxy

Using an SSL certificate will ensure that your data is secure by encrypting the connection to and from Grafana. But, to make use of this connection, you'll first need to reconfigure Nginx.

Open the Nginx configuration file you created when you set up the Nginx server block with Let's Encrypt in the [Prerequisites](#).

```
$ sudo nano /etc/nginx/sites-available/example.com
```

Locate the following block:

```
/etc/nginx/sites-available/example.com  
  
...  
    location / {  
        # First attempt to serve request as file, then  
        # as directory, then fall back to displaying a 404.  
        try_files $uri $uri/ =404;  
    }  
...
```

Because you already configured Nginx to communicate over SSL and because all web traffic to your server already passes through Nginx, you just need to tell Nginx to forward all requests to Grafana, which runs on port 3000 by default.

Delete the existing `try_files` line in this location block and replace it with the following contents, which all begin with `proxy_`.

```
/etc/nginx/sites-available/example.com  
  
...  
    location / {  
        proxy_pass http://localhost:3000;  
        proxy_http_version 1.1;  
        proxy_set_header Upgrade $http_upgrade;  
        proxy_set_header Connection 'upgrade';  
        proxy_set_header Host $host;  
        proxy_cache_bypass $http_upgrade;
```

```
}  
...  
}
```

Once you're done, save the file and close your text editor.

Now, test the new settings to make sure everything is configured correctly.

```
$ sudo nginx -t
```

The output should tell you that the `syntax is ok` and that the `test is successful`. If you receive an error message, follow the on-screen instructions.

Finally, activate the changes by reloading Nginx.

```
$ sudo systemctl reload nginx
```

You can now access the default Grafana login screen by pointing your web browser to `https://example.com`. If you're unable to reach Grafana, verify that your firewall is set to allow traffic on port `443` and then re-trace the previous instructions.

With the connection to Grafana encrypted, you can now implement additional security measures, starting with changing Grafana's default administrative credentials.

## Step 3 — Updating Credentials

Because every Grafana installation uses the same administrative login credentials by default, in this step, you'll update the credentials to improve security.

Start by navigating to `https://example.com` from your web browser. This will bring up the default login screen where you'll see the Grafana logo, a form asking you to enter a **User** and **Password**, a **Log in** button, and a **Forgot your password?** link.



Log in

User

email or username

Password

password

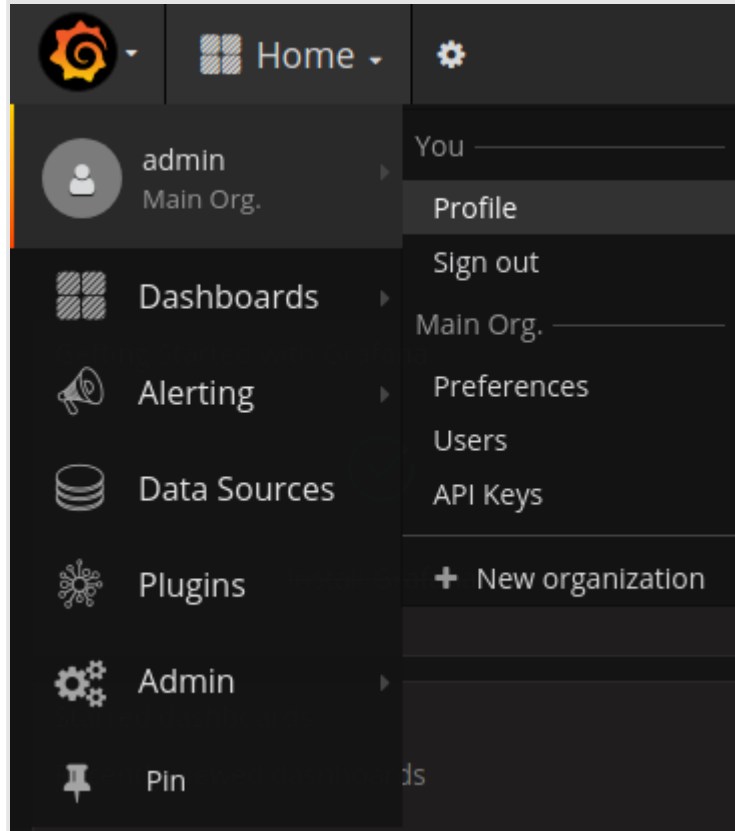
Log in

Forgot your password?

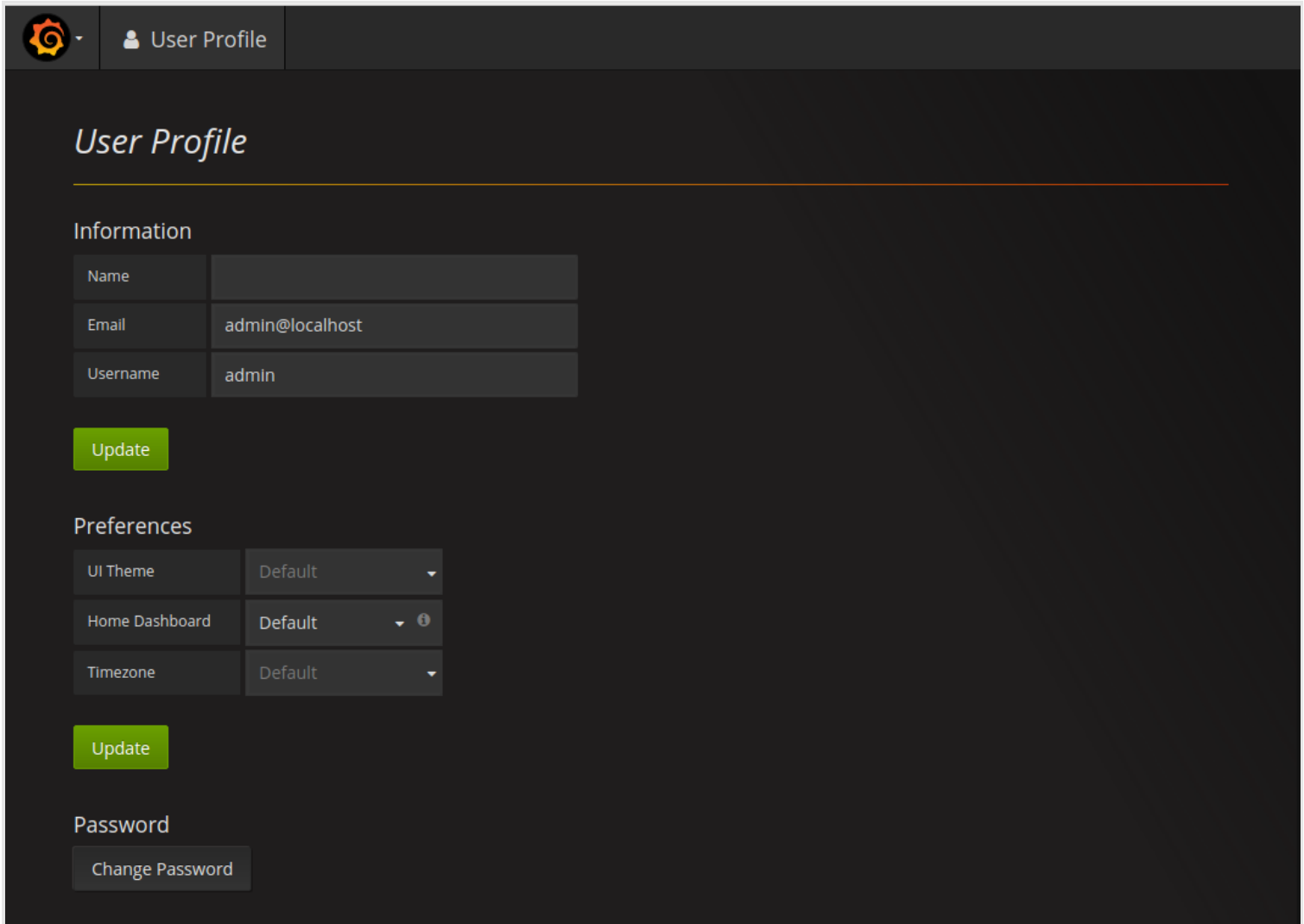
Enter **admin** into both the **User** and **Password** fields and then click on the **Log in** button.

On the next screen, you'll be welcomed to the **Home Dashboard**. Here you can add data sources and create, preview, and modify dashboards.

Click on the small Grafana logo in the upper, left-hand corner of the screen to bring up the application's main menu. Then, hover over the **admin** button with your mouse to open up a secondary set of menu options. Finally, click on the **Profile** button.



You're now on the **User Profile** page, where you can change the **Name**, **Email**, and **Username** associated with your account. You can also update your **Preferences** for settings like the **UI Theme**, and you can change your password.



Enter your name, email address, and the username you want to use in the **Name**, **Email**, and **Username** fields and then click the **Update** button in the **Information** section to save your settings.

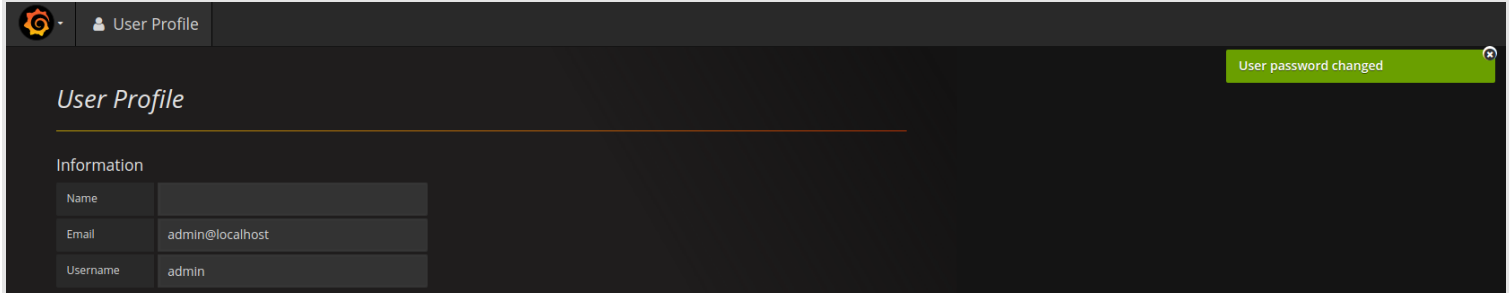
If you want, you can also change the **UI Theme** and **Timezone** to fit your needs and then press the **Update** button in the **Preferences** area to save your changes. Grafana offers **Dark** and **Light** UI themes, as well as a **Default** theme, which is set to **Dark** by default.

Finally, change the password associated with your account by clicking on the **Change Password** button at the bottom of the page. This will take you to the **Change password** screen.

Enter your current password, **admin**, into the **Old Password** field and then enter the password you'd like to start using into the **New Password** and **Confirm Password** fields.

Click **Change Password** to save the new information or press **Cancel** to abandon your changes.

From there, you'll be returned to the **User Profile** page where you'll see a green box in the upper, right-hand corner of the screen telling you that the **User password changed**.



You've now secured your account by changing the default credentials, so let's also make sure that nobody can create a new Grafana account without your permission.

## Step 4 — Disabling Grafana Registrations and Anonymous Access

Grafana provides options that allow visitors to create user accounts for themselves and preview dashboards without registering. As you're exposing Grafana on the internet, this could be a security problem. However, when Grafana isn't accessible via the internet or when working with publicly-available data, like service statuses, you may want to allow these features. So, it's important that you know how to configure Grafana to meet your needs.

Start by opening Grafana's main configuration file for editing.

```
$ sudo nano /etc/grafana/grafana.ini
```

Locate the following `allow_sign_up` directive under the `[users]` heading:

```
/etc/grafana/grafana.ini
```



```
...
[users]
# disable user signup / registration
;allow_sign_up = true
...
```

Enabling this directive with `true` adds a **Sign Up** button to the login screen, allowing users to register themselves and access Grafana.

Disabling this directive with `false` removes the **Sign Up** button and strengthens Grafana's security and privacy.

Unless you need to allow anonymous visitors to register themselves, uncomment this directive by removing the `;` at the beginning of the line and then set the option to `false`.

`/etc/grafana/grafana.ini`

```
...
[users]
# disable user signup / registration
allow_sign_up = false
...
```

Next, locate the following `enabled` directive under the `[auth.anonymous]` heading.

`/etc/grafana/grafana.ini`

```
...
[auth.anonymous]
# enable anonymous access
;enabled = false
...
```

Setting `enabled` to `true` gives non-registered users access to your dashboards; setting this option to `false` limits dashboard access to registered users only.

Unless you need to allow anonymous access to your dashboards, uncomment this directive by removing the `;` at the beginning of the line and then set the option to `false`.

`/etc/grafana/grafana.ini`

```
...
[auth.anonymous]
enabled = false
...
```

Save the file and exit your text editor.

To activate the changes, restart Grafana.

```
$ sudo systemctl restart grafana-server
```

Verify that everything is working by checking Grafana's service status.

```
$ sudo systemctl status grafana-server
```

Like before, the output should report that Grafana is **active (running)** . If it isn't, review any terminal messages for additional help.

Now, point your web browser to <https://example.com> to verify that there is no **Sign Up** button and that you can't sign in without entering login credentials.

If you see the **Sign Up** button or you're able to login anonymously, re-examine the preceding steps to resolve the problem before continuing the tutorial.

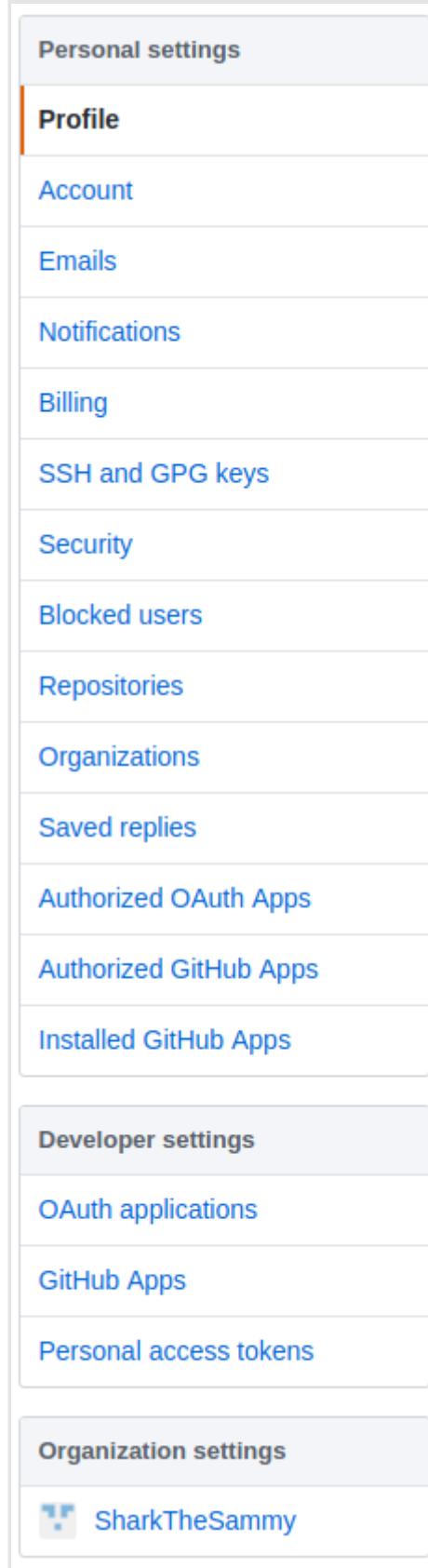
At this point, Grafana is fully configured and ready for use. Optionally, you can simplify the login process for your organization by authenticating through GitHub.

## (Optional) Step 5 — Setting up a GitHub OAuth App

For an alternative approach to signing in, you can configure Grafana to authenticate through GitHub, which provides login access to all members of authorized GitHub organizations. This can be particularly useful when you want to allow multiple developers to collaborate and access metrics without having to create Grafana-specific credentials.

Start by logging into a GitHub account associated with your organization and then navigate to your GitHub profile page at <https://github.com/settings/profile> .


Click on your organization's name under **Organization settings** in the navigation menu on the left-hand side of the screen.




On the next screen, you'll see your **Organization profile** where you can change settings like your **Organization display name**, organization **Email**, and organization **URL**.

Because Grafana uses OAuth — an open standard for granting remote third-parties access to local resources — to authenticate users through GitHub, you'll need to create a new OAuth application within GitHub.

Click the **OAuth Apps** link under **Developer settings** on the lower, left-hand side of the screen.

 This organization Search Pull requests Issues Marketplace Gist

 SharkTheSammy

RepositoriesPeople 1Teams 0Projects 0Settings

Organization settings

ProfileMember privilegesBillingSecurityAudit logBlocked usersWebhooksThird-party accessInstalled GitHub AppsRepository topicsProjects

Developer settingsOAuth AppsGitHub Apps

## Organization profile

Organization display name

Email (will be public)

Description

URL


Location

Billing email (Private)

Gravatar email (Private)

Update profile

Profile picture



Upload new picture

If you don't already have any OAuth applications associated with your organization on GitHub, you'll be told there are **No Organization Owned Applications**. Otherwise, you'll see a list of the OAuth applications already connected to your account.


Click the **Register an application** button to continue.

On the next screen, you'll fill in the following details about your Grafana installation:

- **Application Name** - This helps you distinguish your different OAuth applications from one another.
- **Homepage URL** - This tells GitHub where to find Grafana.
- **Application Description** - This provides a description of your OAuth application's purpose.
- **Application callback URL** - This is the address where users will be sent once successfully authenticated. For Grafana, this field must be set to `https://example.com/login/github`.

Keep in mind that Grafana users logging in through GitHub will see the values you entered in the first three preceding fields, so be sure to enter something meaningful and appropriate.

When completed, the form should look something like:

 This organization Search Pull requests Issues Marketplace Gist

SharkTheSammy

RepositoriesPeople 1Teams 0Projects 0Settings

Organization settings

Profile

Member privileges

Billing

Security

Audit log

Blocked users

Webhooks

Third-party access

Installed GitHub Apps

Repository topics

Projects

Developer settings

**OAuth Apps**

GitHub Apps

## Register a new OAuth application

**Application name**

Grafana

Something users will recognize and trust

**Homepage URL**

https://example.com

The full URL to your application homepage

**Application description**

Authentication for Grafana

This is displayed to all users of your application

**Authorization callback URL**

https://example.com/login/github

Your application's callback URL. Read our [OAuth documentation](#) for more information.

Register application

Cancel

Click the green, **Register application** button.

You will now be redirected to a page containing the **Client ID** and **Client Secret** associated with your new OAuth application. Make note of both values, because you will need to add them to Grafana's main configuration file to complete the setup.

The screenshot shows the GitHub interface for an organization named 'SharkTheSammy'. At the top, there's a navigation bar with links for 'Pull requests', 'Issues', 'Marketplace', and 'Gist'. Below this, a blue banner indicates 'Application created successfully'. The main content area is titled 'SharkTheSammy' and includes tabs for 'Repositories', 'People 1', 'Teams 0', 'Projects 0', and 'Settings' (which is selected). On the left, there's a sidebar with 'Organization settings' (containing Profile, Member privileges, Billing, Security, Audit log, Blocked users, Webhooks, Third-party access, Installed GitHub Apps, Repository topics, and Projects) and 'Developer settings' (containing OAuth Apps and GitHub Apps). The main panel shows the 'Grafana' application settings. It states 'SharkTheSammy owns this application.' with a 'Transfer ownership' button. A note mentions listing the application in the 'GitHub Marketplace' with a corresponding button. Below this, it shows '0 users'. The 'Client ID' and 'Client Secret' are displayed as blurred text, with buttons to 'Revoke all user tokens' and 'Reset client secret'. There's an 'Application logo' section with an 'Upload new logo' button and a 'Drag & drop' area. At the bottom, the 'Application name' is set to 'Grafana'.

**Warning:** Make sure to keep your **Client ID** and **Client Secret** in a secure and non-public location, because they could be used as the basis of an attack.

With your GitHub OAuth application created, you're now ready to reconfigure Grafana.

## (Optional) Step 6 — Configuring Grafana as a GitHub OAuth App

To begin, open the main Grafana configuration file.

```
$ sudo nano /etc/grafana/grafana.ini
```

Locate the `[auth.github]` heading, and uncomment this section by removing the `;` at the beginning of every line, except `;team_ids=`, which we won't be using in this tutorial.

Then, configure Grafana to use GitHub with your OAuth application's `client_id` and `client_secret` values.

- Set `enabled` and `allow_sign_up` to `true`. This will enable GitHub Authentication and permit members of the allowed organization to create accounts themselves. Note that this setting is different than the `allow_sign_up` property under `[users]` that you changed in [Step 4](#).
- Set `client_id` and `client_secret` to the values you got while creating your GitHub OAuth application.
- Set `allowed_organizations` to the name of your organization to ensure that only members of your organization can sign up and log into Grafana.

The complete configuration should look like:

`/etc/grafana/grafana.ini`

```
...
[auth.github]
enabled = true
allow_sign_up = true
client_id = your_client_id_from_github
client_secret = your_client_secret_from_github
scopes = user:email,read:org
auth_url = https://github.com/login/oauth/authorize
token_url = https://github.com/login/oauth/access_token
api_url = https://api.github.com/user
;team_ids =
allowed_organizations = your_organization_name
...
```

You've now told Grafana everything it needs to know about GitHub, but to complete the setup, you'll need to enable redirects behind a reverse proxy. This is done by setting a `root_url` value under the `[server]` heading.

`/etc/grafana/grafana.ini`

```
...
[server]
root_url = https://example.com
...
```

Save your configuration and close the file.

Then, restart Grafana to activate the changes.

```
$ sudo systemctl restart grafana-server
```

Lastly, verify that the service is up and running.

```
$ sudo systemctl status grafana-server
```

If the output doesn't indicate that the service is `active (running)` , consult the on-screen messages for more information.

Now, test your new authentication system by navigating to `https://example.com` . If you are already logged into Grafana, click on the small Grafana logo in the upper, left-hand corner of the screen, hover your mouse over your username, and click on **Sign out** in the secondary menu that appears to the right of your name.

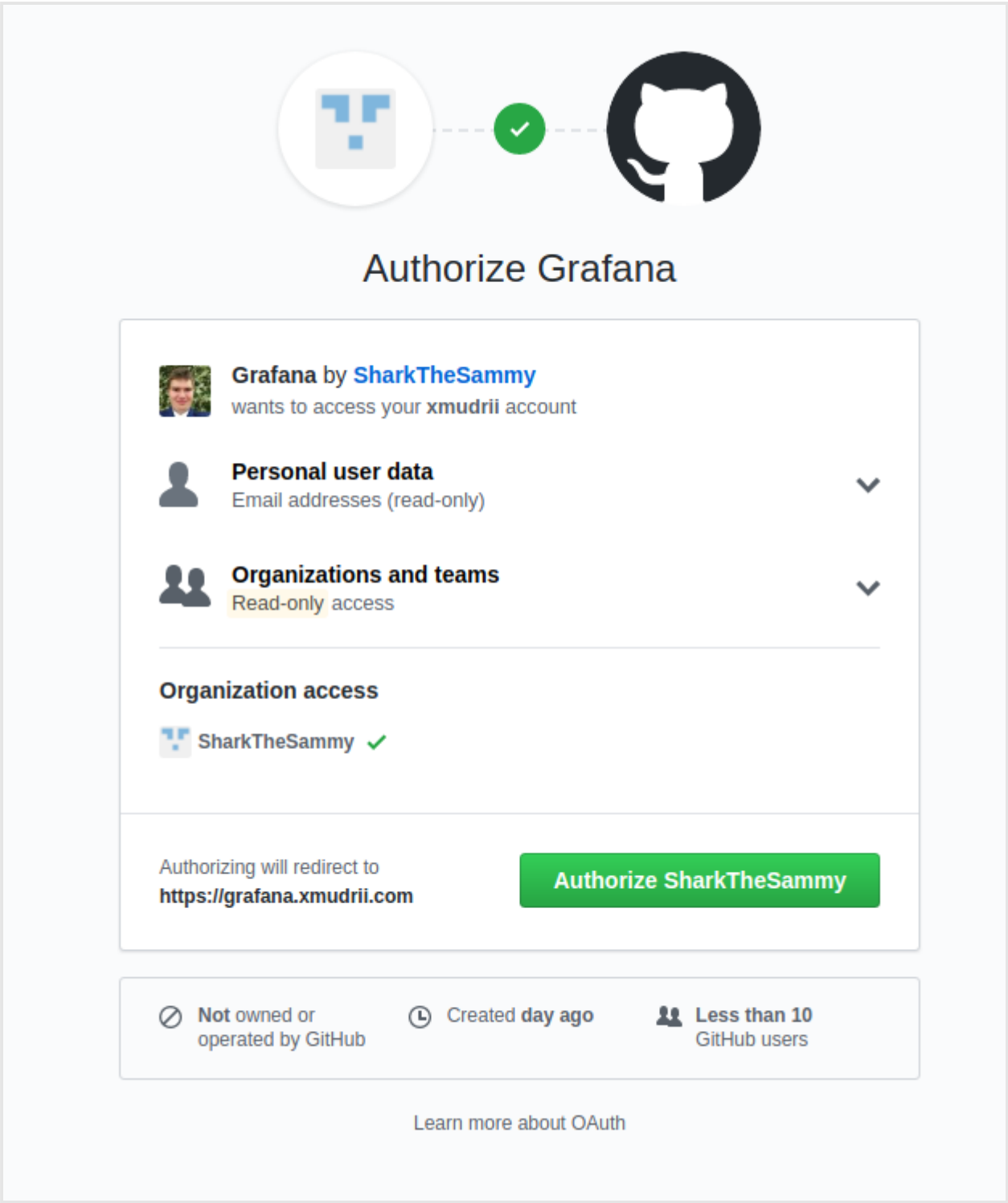
On the login page, you'll see a new section under the original **Log in** button that includes a **GitHub** button with the GitHub logo.



Click on the **GitHub** button to be redirected to GitHub, where you'll need to confirm your intention to **Authorize Grafana**.



Click the green, `Authorize your_github_organization` button. In this example, the button reads, **Authorize SharkTheSammy**.



If you try to authenticate with a GitHub account that isn't a member of your approved organization, you'll get a **Login Failed** message telling you, **User not a member of one of the required organizations**.

If the GitHub account is a member of your approved organization and your Grafana email address matches your GitHub email address, you will be logged in with your existing Grafana account.

But, if a Grafana account doesn't already exist for the user you logged in as, Grafana will create a new user account with **Viewer** permissions, ensuring that new users can only use existing dashboards.

To change the default permissions for new users, open the main Grafana configuration file for editing.

```
$ sudo nano /etc/grafana/grafana.ini
```

Locate the `auto_assign_org_role` directive under the `[users]` heading, and uncomment the setting by removing the `;` at the beginning of the line.

Set the directive to one of the following values:

- **Viewer** — can only use existing dashboards
- **Editor** — can change use, modify, and add dashboards
- **Admin** — has permission to do everything

```
                                /etc/grafana/grafana.ini

...
[users]
...
auto_assign_org_role = Viewer
...
```

Once you've saved your changes, close the file and restart Grafana.

```
$ sudo systemctl restart grafana-server
```

Check the service's status.

```
$ sudo systemctl status grafana-server
```

Like before, the status should read `active (running)`. If it doesn't, review the output for further instructions.

At this point, you have fully configured Grafana to allow members of your GitHub organization to register and use your Grafana installation.

## Conclusion

In this tutorial you installed, configured, and secured Grafana, and you also learned how to permit members of your organization to authenticate through GitHub.

To use Grafana as part of a system-monitoring software stack, see [How To Install Prometheus on Ubuntu 16.04](#) and [How To Add a Prometheus Dashboard to Grafana](#).

To extend your current Grafana installation, see the [list of official and community-built dashboards](#).

And, to learn more about using Grafana in general, see the [official Grafana documentation](#).



Editor:  
Dave Rankin



We just made it easier for you to deploy faster.

[TRY FREE](#)

### Related Tutorials

How To Set Up Jupyter Notebook with Python 3 on Ubuntu 18.04

How To Set Up a Jupyter Notebook with Python 3 on Debian 9

How To Install the Anaconda Python Distribution on Debian 9

How To Install the Anaconda Python Distribution on Ubuntu 18.04

How To Spin Up a Hadoop Cluster with DigitalOcean Droplets

## 6 Comments

Leave a comment...

Log In to Comment

^ [TomKeen](#) January 5, 2018



0 Great write up. Thank you for writing such a detailed post!

^ [MahadevGouda](#) July 4, 2018



0 It is a nice tutorial and explained great.

I have a web application which has a login page and it returns me lot of reports . To dashboard those data I am using Grafana. I want to integrate SSO between my app and grafana. When user login's to my web application he should be logged into grafana too. To do this I went through grafana documentation. I didn't understand much. I tried with google.auth but is not right way for my requirement. It should be possible to login using the credentials which are used to login to my web application. Any work around for this??

I also went through authproxy documentation and this is what I tried

```
[auth.proxy]
enabled = true
header_name = X-WEBAUTH-USER
header_property = username
auto_sign_up = true
ldap_sync_ttl = 60
whitelist =
```

```
curl -H "X-WEBAUTH-USER: anthony" http://localhost:3000/api/user
```

But user has not logged in to the grafana. Can you let me know what mistake did I do?? and what did I miss understand.

^ [xMudrii](#) July 4, 2018

1 The AuthProxy seems like the best way to accomplish this. If your application supports OAuth, you can try the Generic OAuth as well. But I believe, you'll mostly like have more luck with AuthProxy.

I don't have experience with AuthProxy, but as far I understand from the documentation, the curl call is not going to login you to Grafana. The `/api/users/` endpoint is used to list all users and the `/api/user` endpoint can be used to create a new user.

It seems like you need to pass the **X-WEBAUTH-USER** header every time, therefore changes to Apache, or to Nginx in case of the tutorial, are required. Then you should be able to use the modified server block/endpoint to authenticate using the AuthProxy.

I don't have Nginx server handy right now, but if needed, I can try to make similar server block for Nginx. :)

---

^ MahadevGouda July 5, 2018

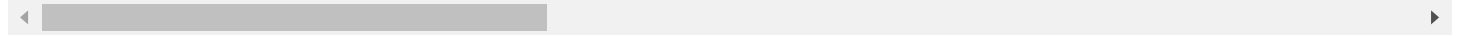


0 @xMudrii

Thanks a lot for your answer

I am not clear about this part

It seems like you need to pass the X-WEBAUTH-USER header every time, therefore changes to Ap



Can you please explain more or share some of the documents or explain the configuration.  
this will be helpfull for me.

---

^ xMudrii July 5, 2018



- 0 The [AuthProxy documentation](#) has instructions what you need to get done on the web server in order to utilize the AuthProxy. However, those instructions are for Apache, and you need instructions for Nginx. Anyways, you can read it to get a sense how it works, as they explained it very well.

The following resources shows how the Nginx server block should look like in order to utilize the AuthProxy. I hope you'll find those helpful:

- [Grafana Auth proxy Does not let the user log in automatically](#) – GitHub.
- [How to use auth proxy with nginx?](#) – Grafana Community.

Following those resources you need to modify the Grafana Nginx server block to use the AuthProxy.

The server block is located in the `/etc/nginx/sites-available/example.com`, where `example.com` is your domain, or name you chose when installing Grafana (that's the same file from Step 2 of this tutorial). You can open it in any text editor, such as:

```
$ sudo nano /etc/nginx/sites-available/example.com
```

Give it a try and let me know does it work for you. If Nginx changes don't work, make sure the check the error logs, which are usually located in the `/var/log/nginx/error.log` file.

---

^ MahadevGouda July 9, 2018



- 0 Thanks for your response,

This what I tried ([my comments -Mahadev](#))

Can you help me out.



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



Copyright © 2019 DigitalOcean™ Inc.

[Community](#) [Tutorials](#) [Questions](#) [Projects](#) [Tags](#) [Newsletter](#) [RSS](#) 

---

[Distros & One-Click Apps](#) [Terms, Privacy, & Copyright](#) [Security](#) [Report a Bug](#) [Write for DOnations](#) [Shop](#)