Dashboard (Grafana)

The FileWave dashboard is a collection of custom code and open-source solutions. We'll give you the beginner's guide here for customizing your dashboard up to some more advanced topics.

FileWave Dashboard Intro

What

The FileWave Dashboard provides summary information about your FileWave server, your deployments, and pretty much anything else you'd like to know about your FileWave environment.

When/Why

The best time to use the dashboard is when summary data is important to you, particularly regarding server performance and deployment progress. It is a great tool to communicate with senior management, or to simply keep a watchful eye on your environment.

How

We won't go into detail in this article about using the Dashboard, but we will do that in all of the great articles in this section of the KB. That said, the FileWave dashboard is a collection of custom code and open-source solutions. We'll give you the beginner's guide here for customizing your dashboard up to some more advanced topics, but if you really want to understand how things work, you'll want to take a look at some of the following resources as well:

- Grafana (graphing engine): https://grafana.com/docs/grafana/latest/
- Prometheus (data scraping/monitoring): https://prometheus.io/docs/

1.0 Dashboard Basics

In this section, we'll take a look at basic access to the FileWave dashboard and give information on it's various built-in components. If you intend to use this dashboard, but you are new to it, then this is the place to start.

1.1 Accessing the FileWave Dashboard

What

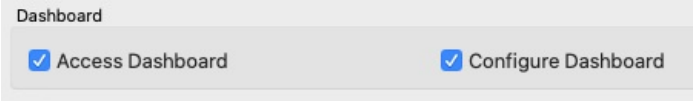
Your account will need permission to access the FileWave Dashboard.

When/Why

There are three permission levels for the dashboard for each admin logon:

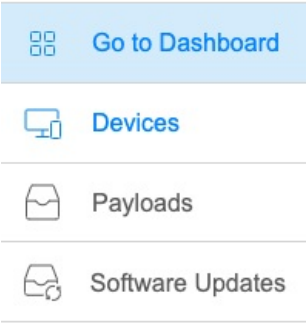
- No Access
- Read-Only Access
- Read-Write Access

The permissions are defined with the following options in the Manage Administrators Assistant:



How

Accessing the dashboard itself is quite simple once your account has proper permissions. From the WebAdmin, simply choose the Go To Dashboard link:

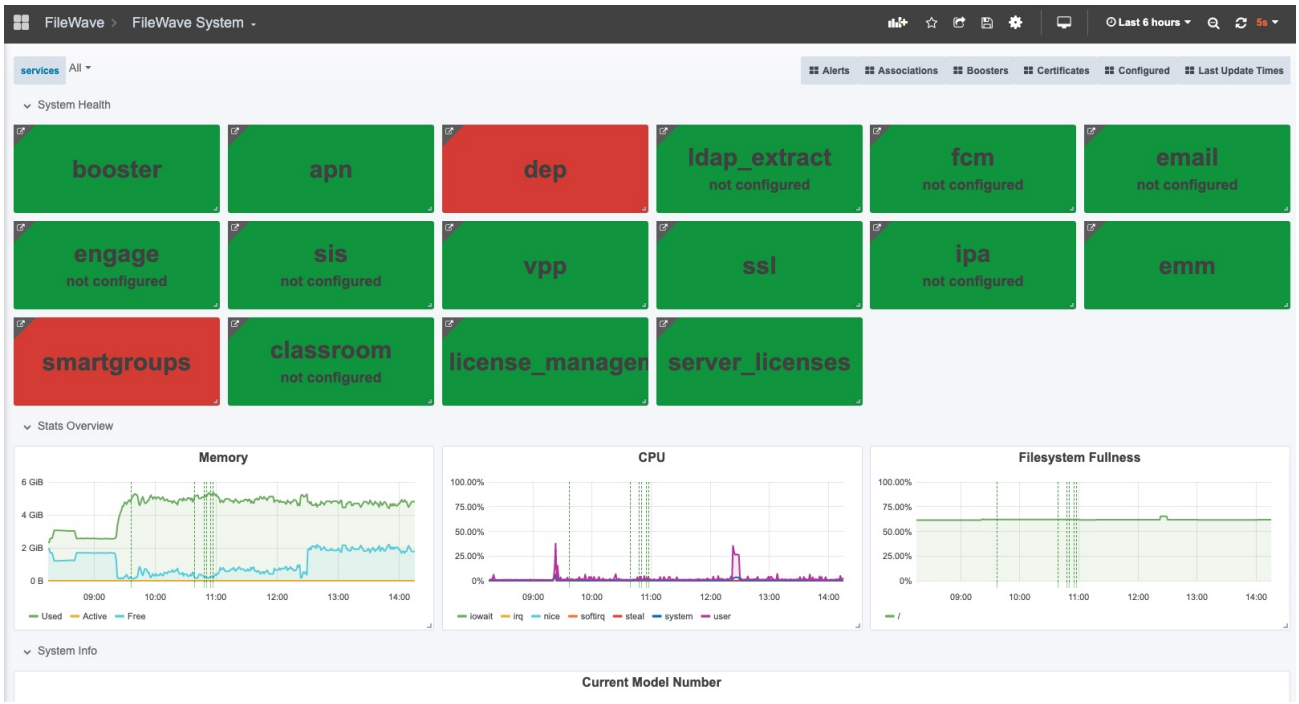


You will automatically be authenticated using the already established credentials to the WebAdmin. Note that all access to the dashboard is routed through the WebAdmin

1.2 Default Dashboard

What

Once you access the dashboard, you are going to be presented with a default system dashboard that looks something like this:



When/Why

This simple dashboard (called FileWave System) is basically the equivalent of the dashboard in the native admin. It shows you basic configuration elements, and let's you know whether there are configuration issues with a simple color code.

How

Taking action to remediate issues will resolve the reporting. i.e. In the above, I do not have a macOS client specified for use by DEP, therefore the DEP widget shows as red. If I uploaded the pre-configured client in my admin, this issue would be resolved, and the widget would turn green.

1.3 FileWave Provided Dashboards

What

FileWave provides a number of default dashboards, some of which will be useful to you directly, and others that will be more useful to support staff.

When/Why

Of course everything that we do with the dashboard is about reporting. There are hundreds of things you might want to report on...we give you many examples to work with which you'll see below:

"Operations" related dashboards are all about the performance of your FileWave systems. "FileWave" bucketed dashboards primarily give information about configuration of the various systems. And the "Patching" group of dashboards is an example of DIY dashboard reporting on patch, application, and deployment status.

How

The important thing to note is that these dashboards are starting points for you, not necessarily destinations unto themselves. You can slice/dice/transform and re-imagine all of these elements in your own dashboards at your discretion. Note that the FileWave provided dashboards can not be modified directly, but they can be copied to dashboards of your own.

1.4 Switching Between Dashboards

What

Switching between various dashboards is the first thing you are going to want to do once you login.

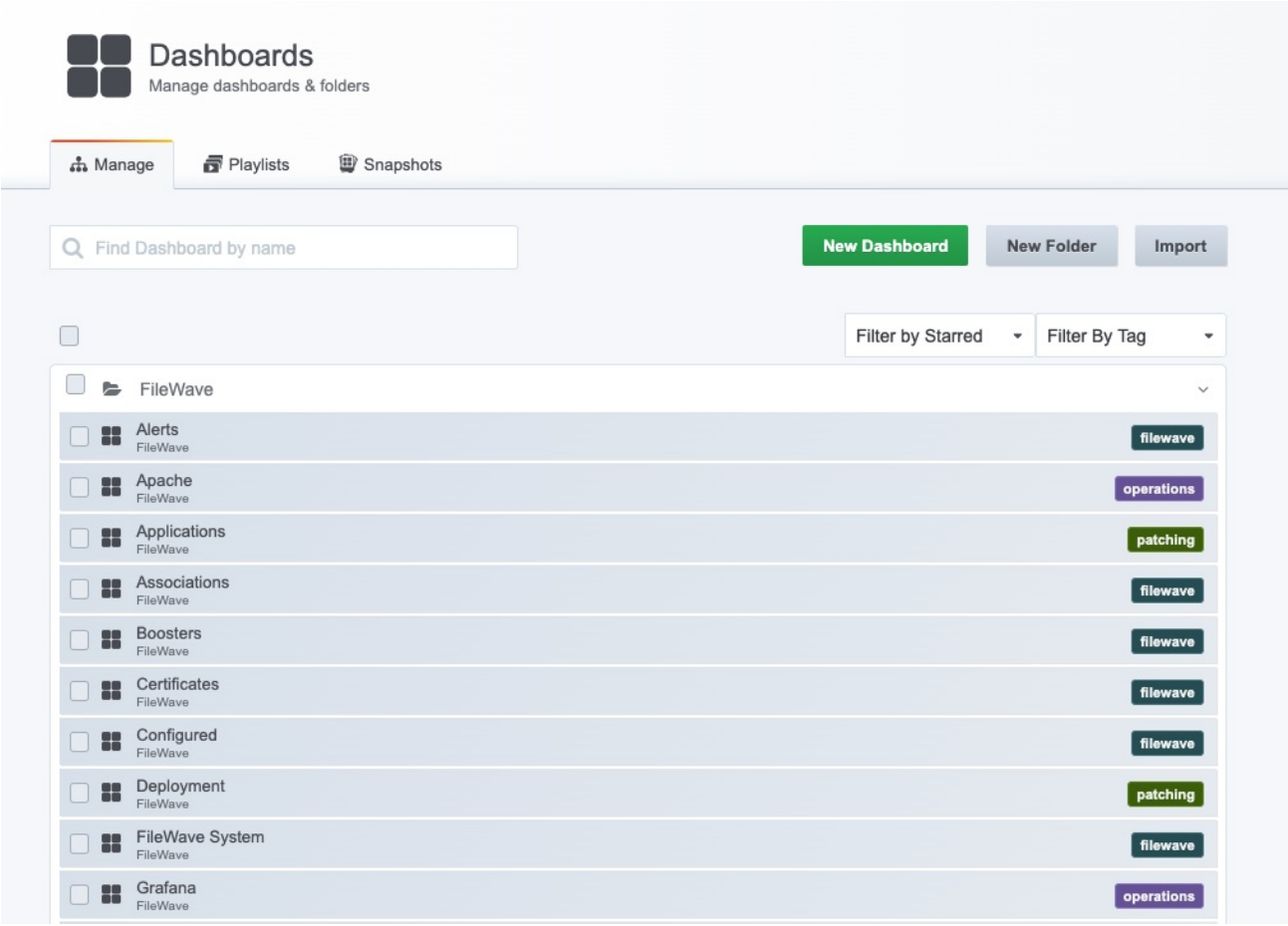
When/Why

Switching between various dashboards allows you to look at different types of data in different ways. I may want to open new browser tabs with different content, I may want to switch between views in the same tab. Or, I may even want the singular tab to cycle through content for me on its own (called a playlist).

How

We'll start simple here, and just discuss switching between different dashboards. The easiest way is to simply use the header links to switch as shown below:

Note that there are other methods too...such as Dashboards→Manage:


Dashboards
Manage dashboards & folders
Manage
Playlists
Snapshots
Find Dashboard by name
New Dashboard
New Folder
Import
Filter by Starred
Filter By Tag
FileWave
Alerts
FileWave
filewave
Apache
FileWave
operations
Applications
FileWave
patching
Associations
FileWave
filewave
Boosters
FileWave
filewave
Certificates
FileWave
filewave
Configured
FileWave
filewave
Deployment
FileWave
patching
FileWave System
FileWave
filewave
Grafana
FileWave
operations

1.5 Dashboard Panel/Widget Layout

What

The dashboard panels are highly customizable and allow you to change the appearance of your dashboard to suit your needs.

When/Why

Just copying and pasting content into a dashboard is a good first step, but we can tailor the information to be portrayed in just the way we like.

How

Simply put, every panel/widget can be resized and moved around to your chosen location, as shown below:

1.6 Exposing an Association to Dashboard (v14+)

What

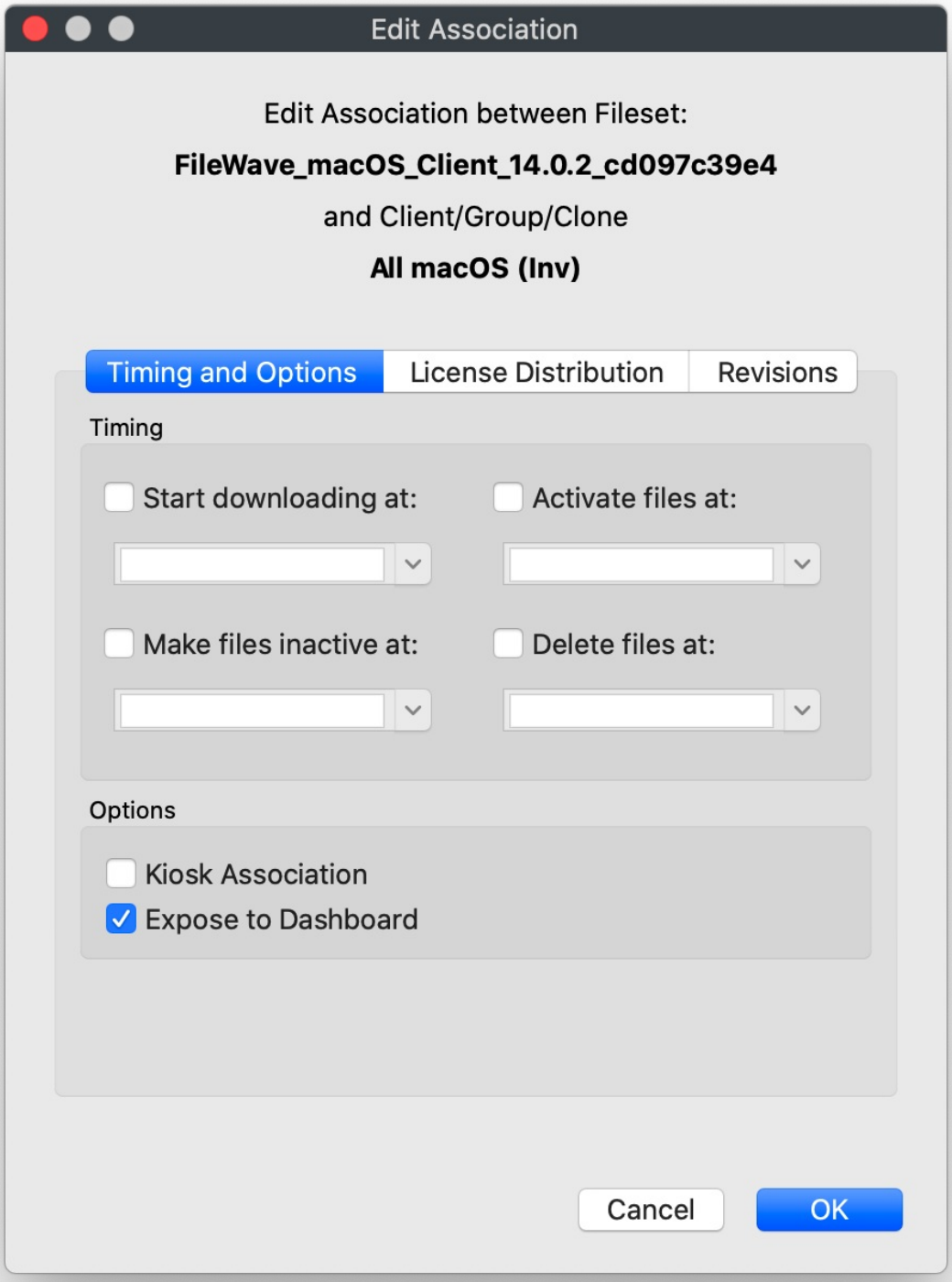
With the new FileWave Dashboard, it is possible to track the status of any deployment graphically.

When/Why

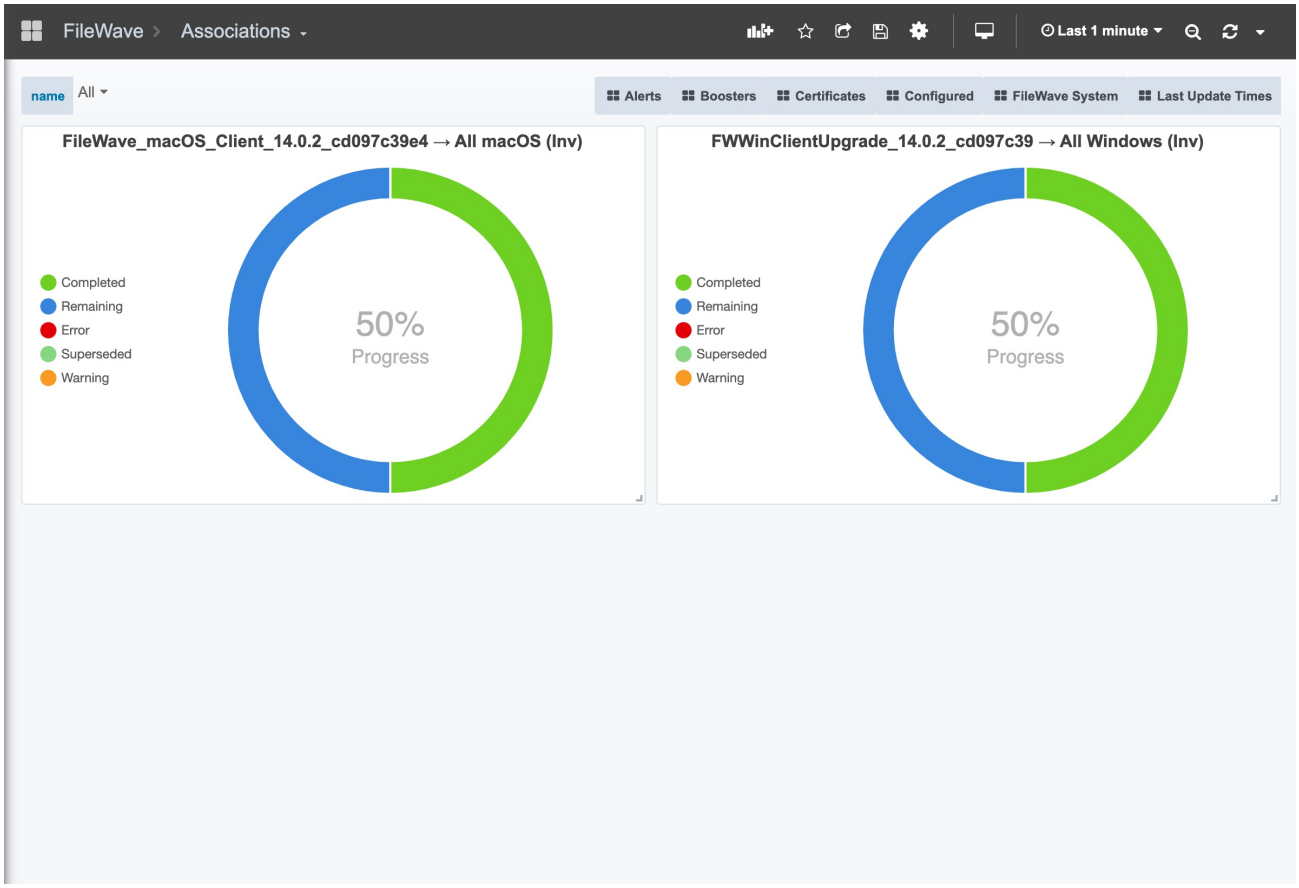
This will be important to us anytime we have a significant rollout, and especially if we need to communicate the status to others in the organization.

How

To make the data show, simply select the "Expose to Dashboard" option on any association as shown below:



Once these changes are confirmed, the dashboard will show deployment status, similar to the below:



Note that for clients to send data to populate these charts, they must be at a minimum version of 14.

Importing a Grafana Dashboard

What

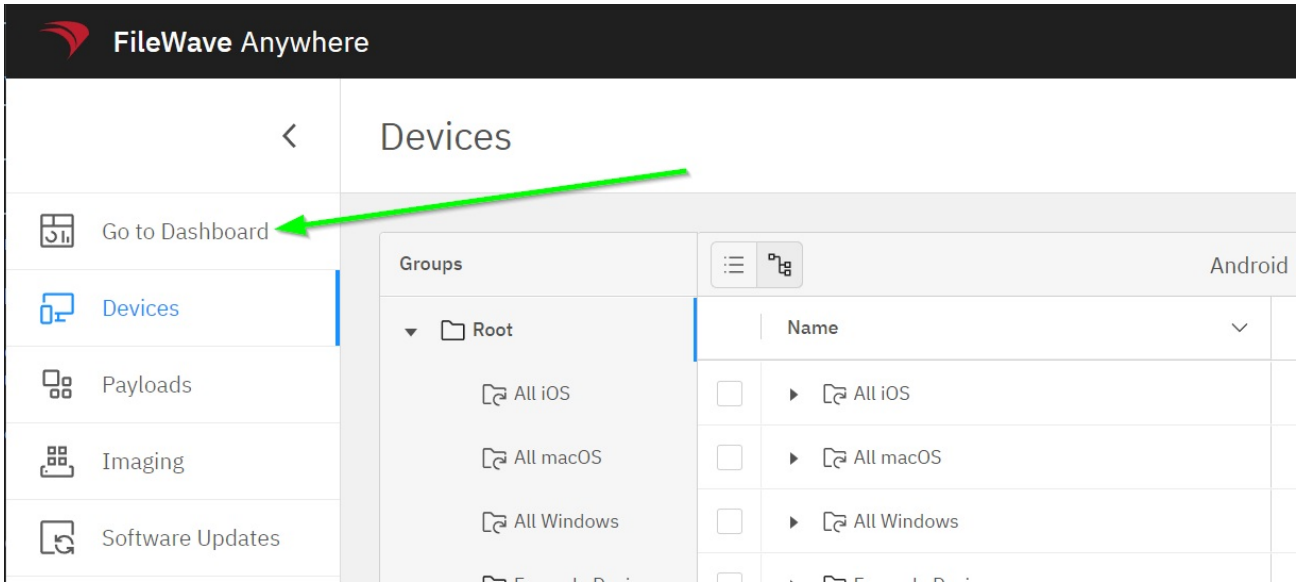
There's no need to build a grafana dashboard from scratch if you can "borrow" one from a friend. This article explains how to import a grafana dashboard from another system.

When/Why

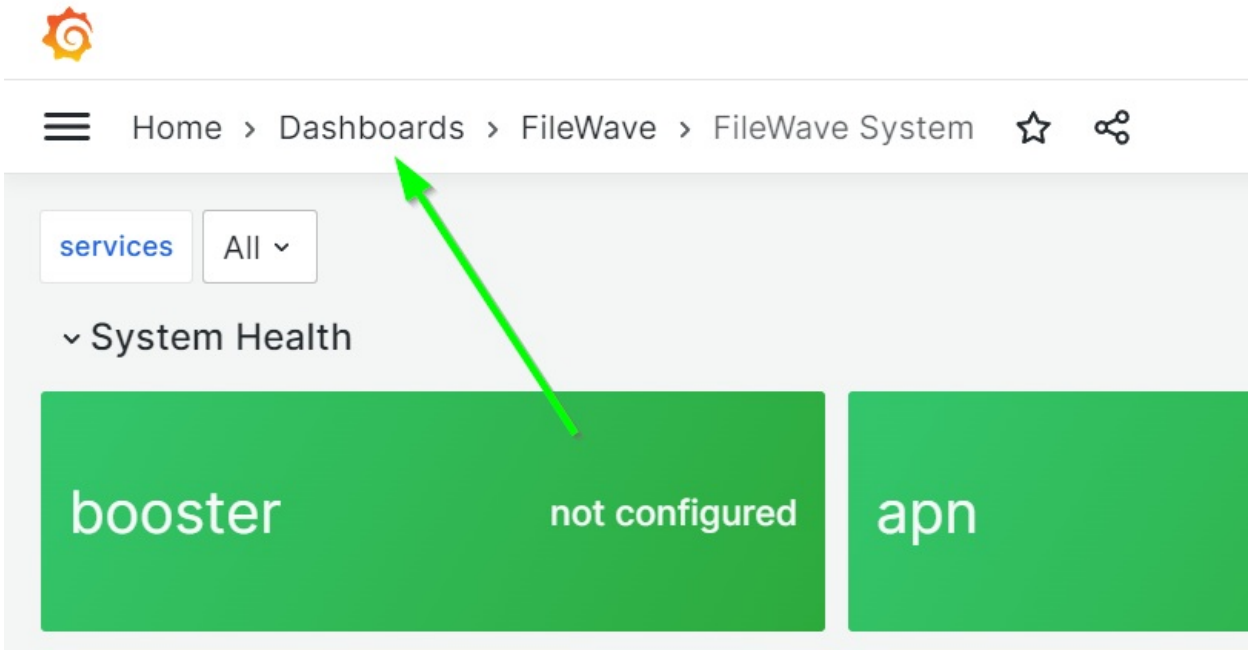
We'll want to import a dashboard whenever someone else has done the work for us, and we'd like to have a dashboard the easy way.

How

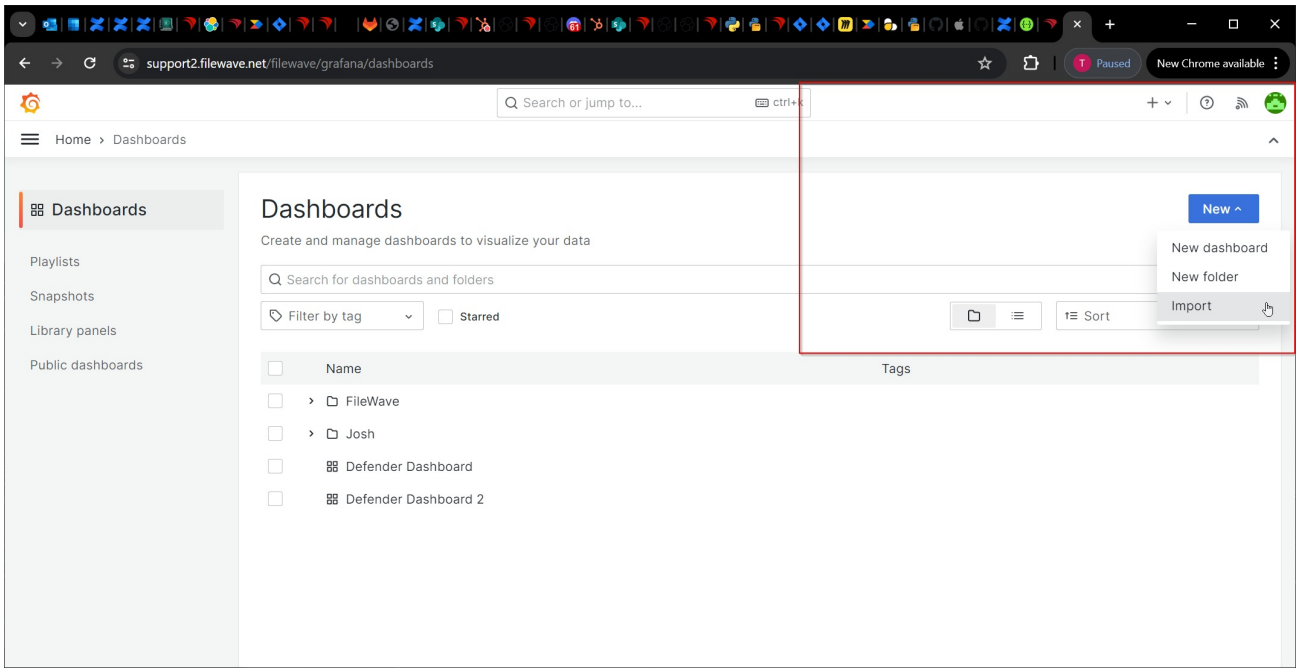
The steps are actually quite simple...from FileWave Anywhere, Go To Dashboard:



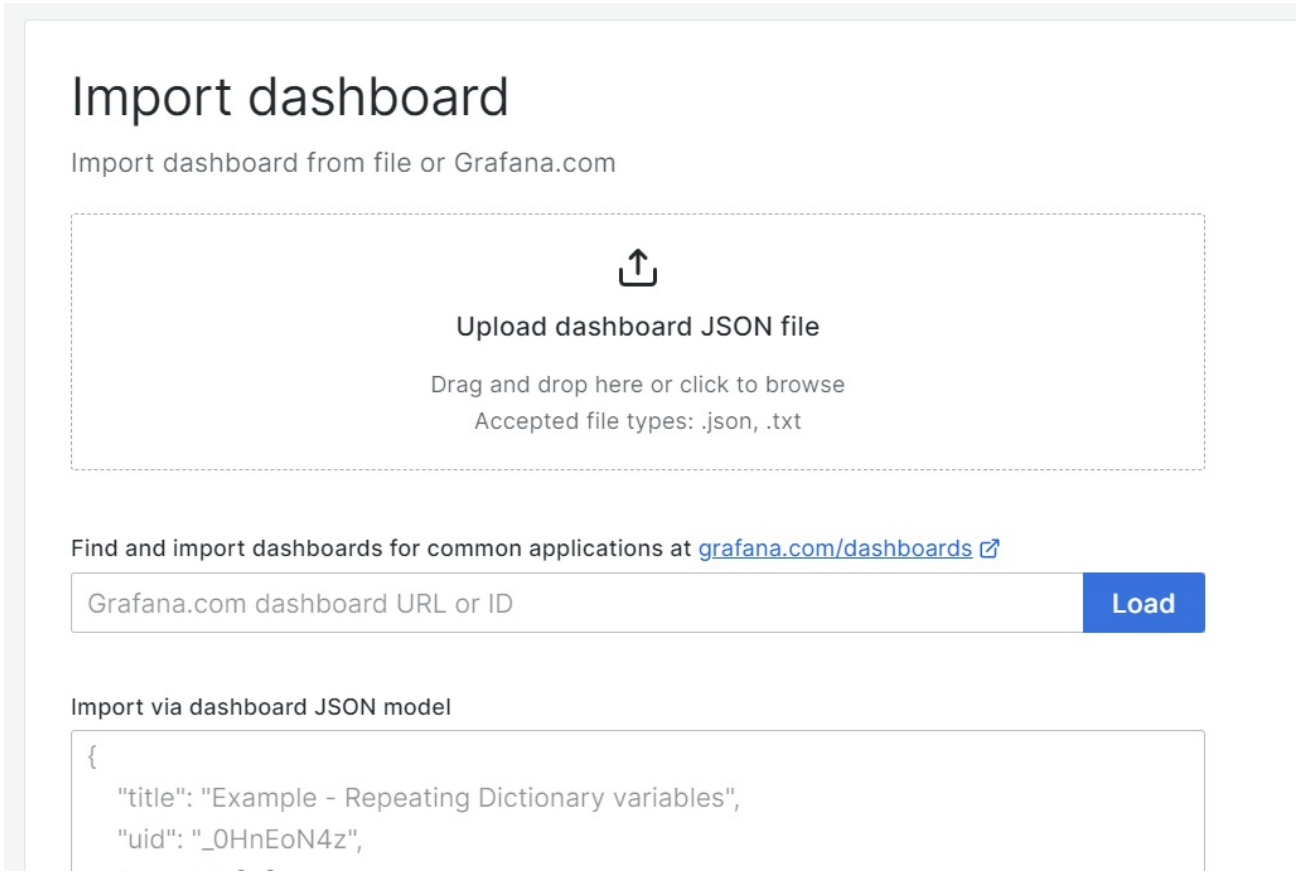
Within the dashboard, go to Home> Dashboards:



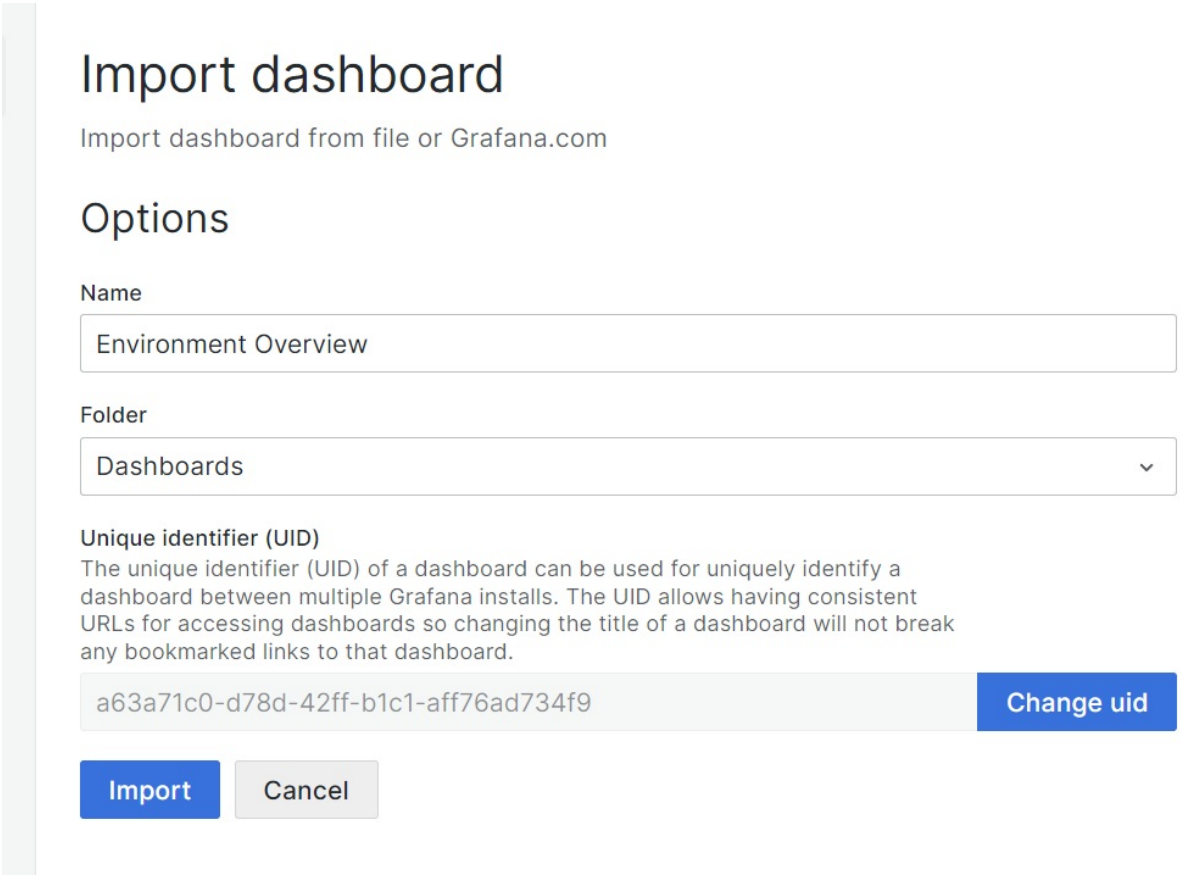
Then to New, Import:



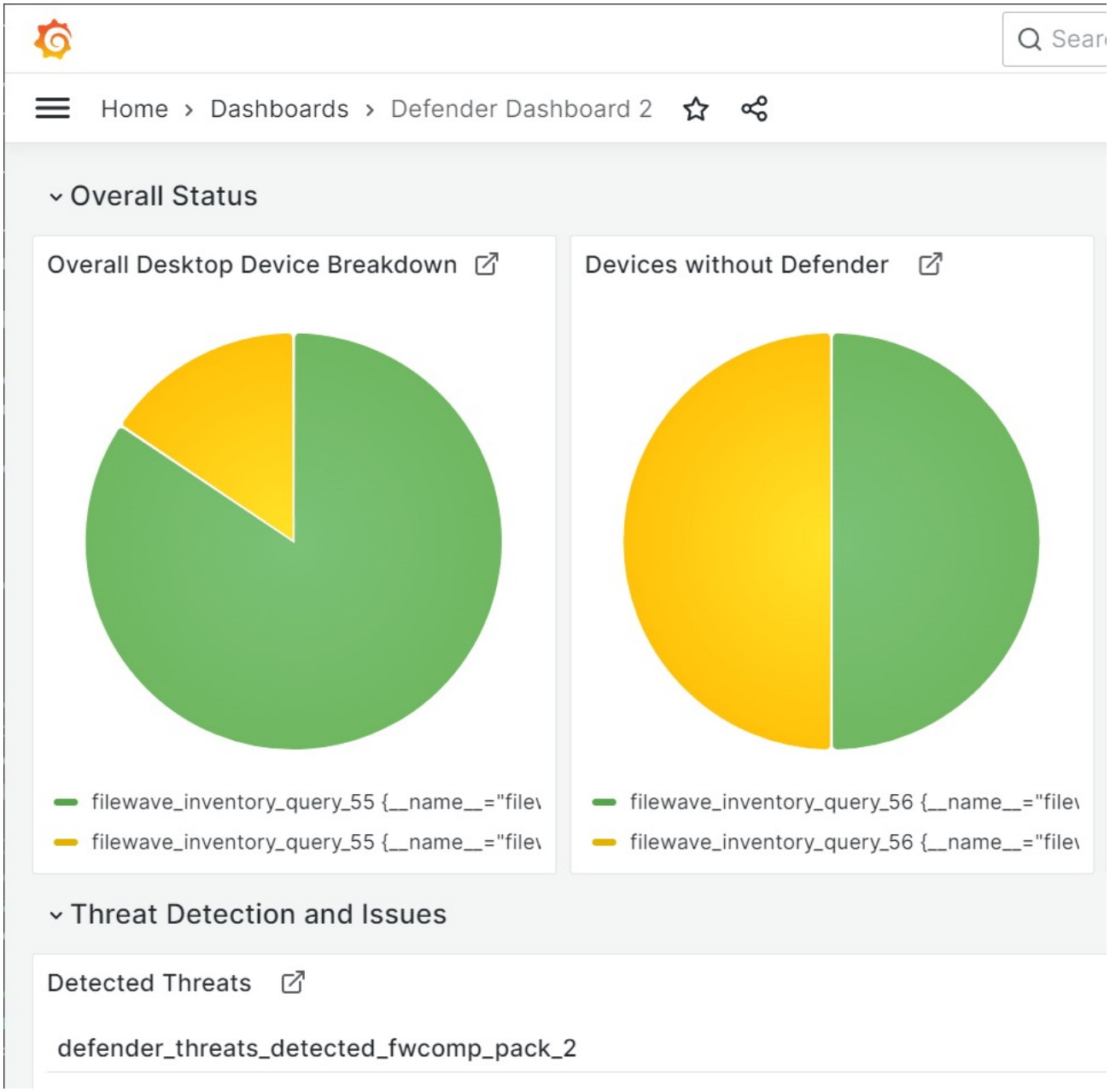
And then either import the JSON file or copy and paste the JSON content:



Now just set a name and a destination (note: every dashboard must have a unique UID):



And that is it, our new dashboard is imported:



Related Content

- Content Packs
- Dashboard (Grafana)

2.0 Custom Dashboards for Beginners

Once you get the hang of the elements that are provided for you automatically, you may want to take the next step and start creating custom dashboards of your own.

FileWave supplied dashboards will only take you just so far. As soon as you want to know specific information about your environment and your deployments you are going to want to create your own dashboards with your own dashboard elements.

The contents for customization are broken into two parts. In the first (this section) we'll look at the components of dashboards and show you how you can build your own simple custom dashboards. In the next section, we'll get much more detailed and advanced.

Using Grafana for Data Aggregation

What

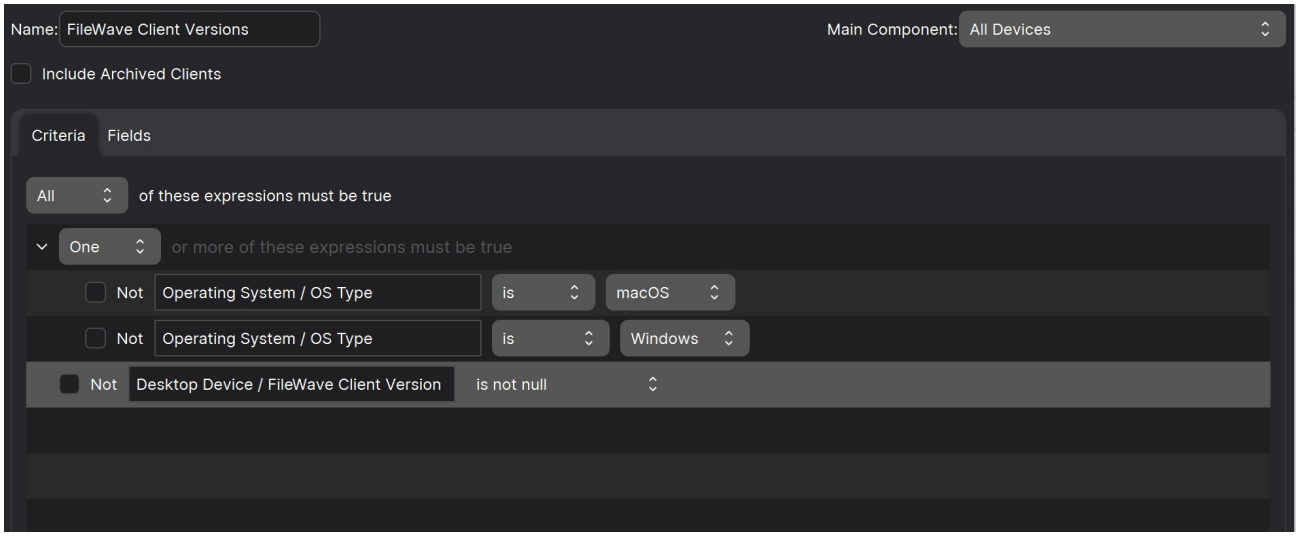
Grafana has been included within FileWave for quite some time, but only with a recent update does Grafana have the ability to do data aggregation without using Prometheus. This article shows you how to use this feature.

When/Why

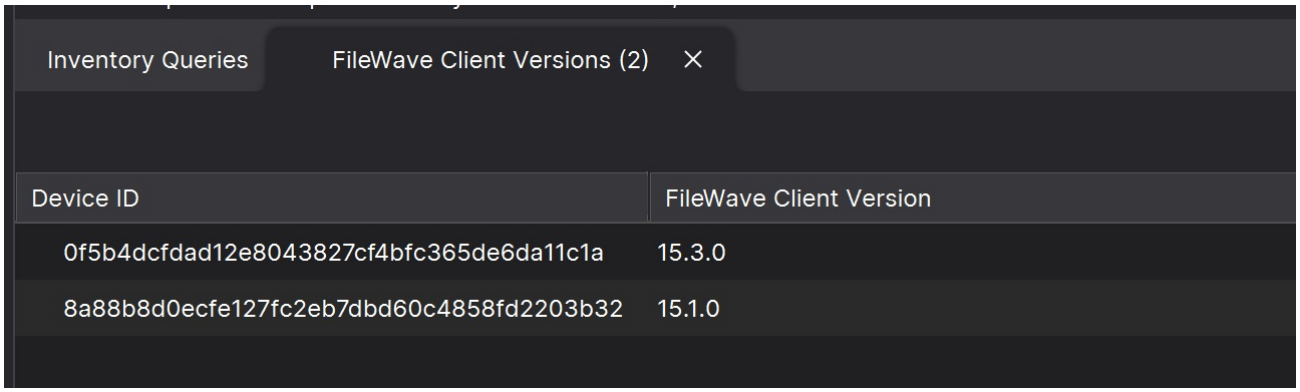
We'll use Data Aggregation whenever we want to look at an overview of data. For instance, if we want to understand how many devices are missing OS patches, we might create a report showing all devices, and their missing patches. To give a graphical representation of this, we would likely Group this data by patch name, and Count the number of devices missing each. This data is perfect for a visualization like a bar or pie chart.

How

To create an aggregated visualization, we need to start with a report in FileWave. In this instance, we'll create a report on the version of the FileWave client on Mac and Windows devices:



Note that we included that the client versions is not null...this prevents having placeholders, etc in the data. Also note that we only included two fields here: FileWave Client Version (which we intend to Group By), and Device ID (which we intend to count). There aren't may devices in this system, so the data just looks like this:



Now we are ready for the "dashboard" part of the exercise, so go into your Dashboard from FileWave Anywhere and follow along with this short video:

Related Content

- Dashboard (Grafana)
- Content Packs

2.1 Creating Your Own Dashboard

What

The beauty of the FileWave dashboard isn't in what we give you...although we definitely give you some great stuff! The beauty of the solution is that you can make it what you want it to be using your own dashboards.

When/Why

Dashboard elements (or widgets as we will call them) can be combined to create dashboards of your own. You can copy these widgets from pre-existing FileWave Dashboards, or even create your own widgets directly.

How

In this example, we'll just look at creating the new empty dashboard we want, but do make sure and follow this up with the article below on copying widgets. Note that we create a new folder first, as we expect we'll create more than one dashboard eventually:

2.2 Copying Widgets (Panels)

What

If we have the power to create our own dashboards, then surely we must be able to add content to them...

When/Why

We are going to copy a widget (referred to as a panel in Grafana) whenever one already exists and we can leverage it without doing any extra work. For instance, watch below, as we "steal" content from various pre-existing dashboards for our own custom dashboard.

How

Taking pre-existing content from other dashboards is as easy a copy and paste (rinse and repeat):

If you find that your copied panel is blank, take a look at resolving that in this article: Fixing Blank Copied Panel

2.2.1 Copied Panel is Blank

What

In certain circumstances, when you copy/paste a panel into another dashboard, you may find that the new panel is blank.

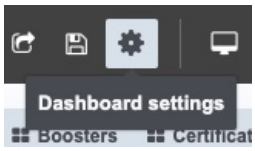
When/Why

This will happen most often if the dashboard that you copied from has variables defined that the panel relies upon. For instance, if a panel relies on $host to be the address of your server, but the destination dashboard doesn't have that variable, then the panel will copy, but be blank.

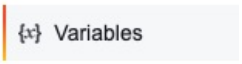
How

Below you'll see the issue:

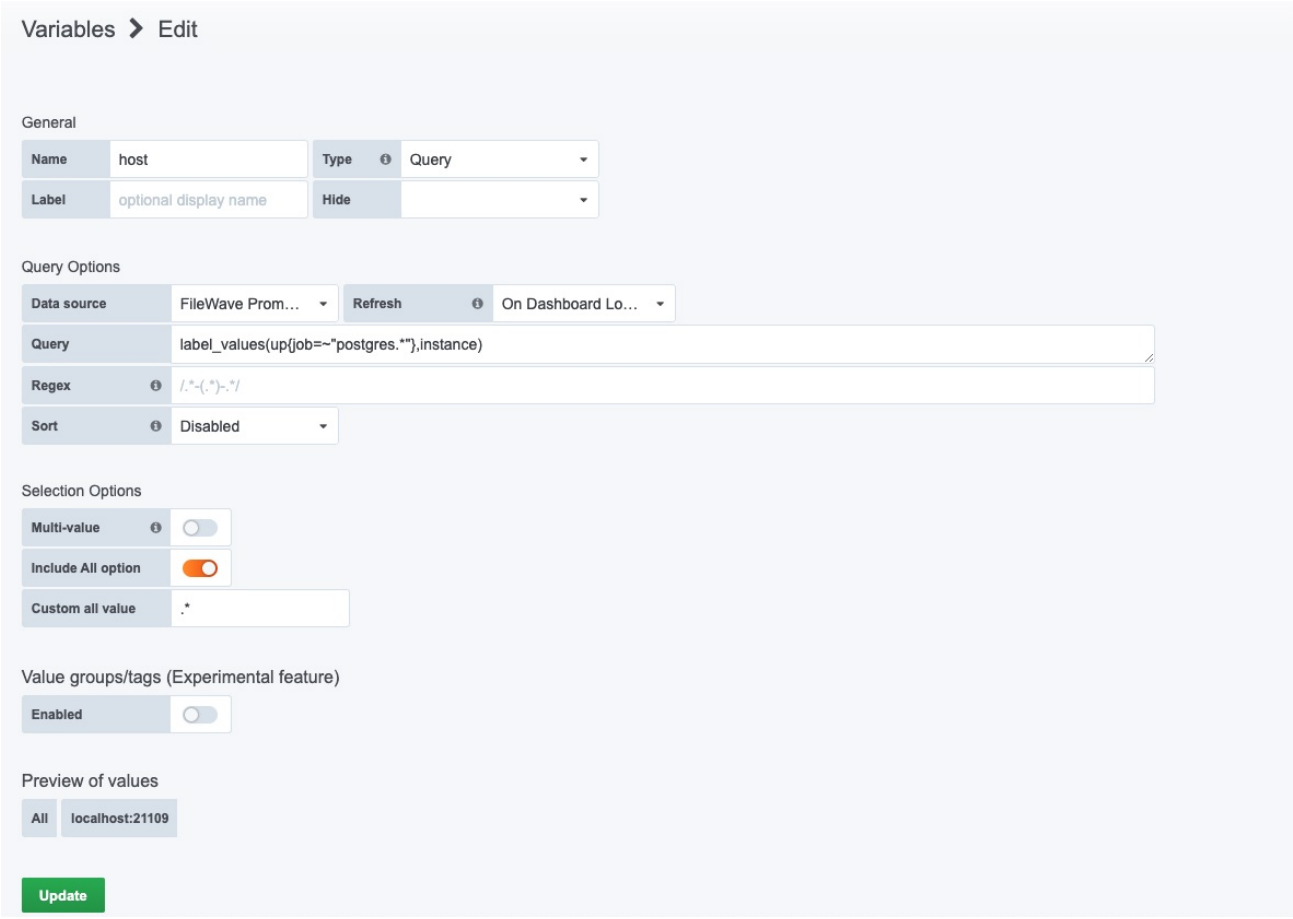
Fixing the issue requires replicating the variables in the destination dashboard. (Unfortunately there is no method for copy/paste, so they must be regenerated.). In any dashboard, you can see the variables in Dashboard Settings → Variables:



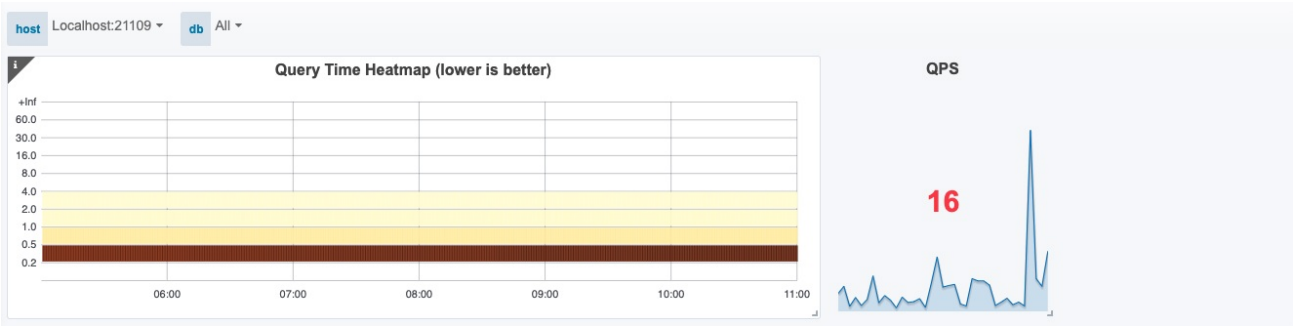
then



Replicate the variables in your own dashboard:



And, once you save those new variables, you should find your copied panel now works:



2.3 Widget/Panel Elements

What

All panels (or widgets) on the FileWave dashboard are comprised of the same basic elements. This article reviews those elements at a high level.

When/Why

If you are using pre-existing panels, you won't care too much about how they are built. However, as soon as you want to build your own panels, the building blocks become quite important.

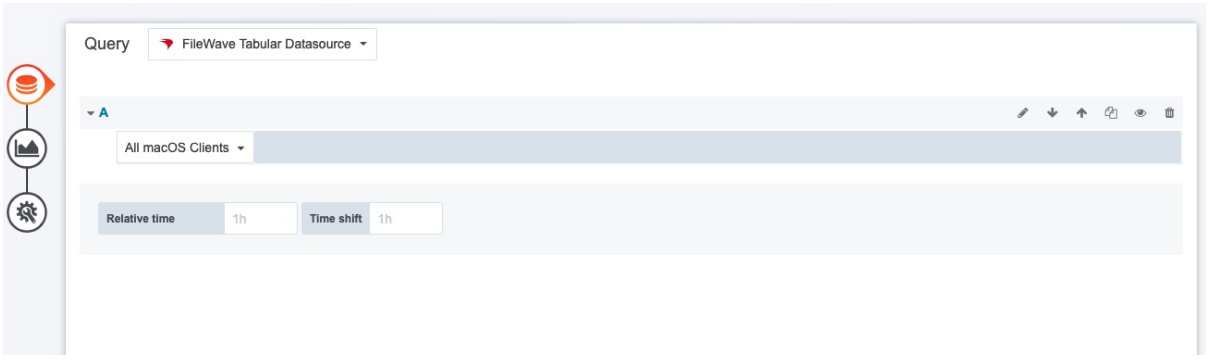
How

All panels have the following three elements:

- The Query (Choosing what data you want to see in the panel)
- The Visualization (Choosing the visual representation of the data: gauge, bar chart, table...)
- General (Panel properties such as title, comment, links)

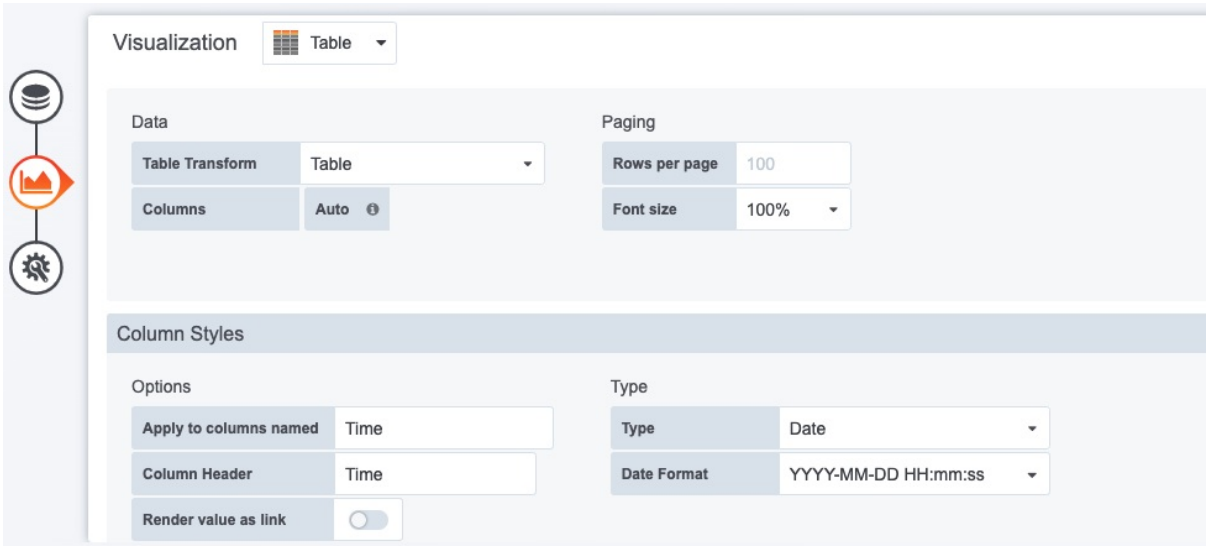
In the below you'll see examples of each of these elements:

The Query



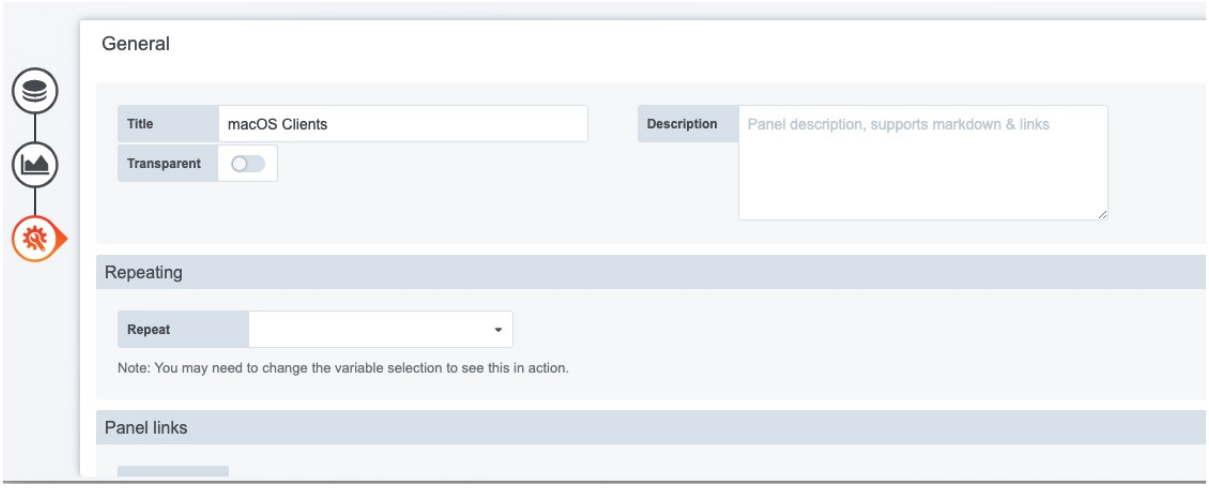
The query you choose defines the data provided to the panel. In this case, an inventory query was selected.

The Visualization



Because we chose an inventory query as our data source, a table was our only possible visualization. In our more advanced example though, you'll see that we can do charts and graphs here of all sorts.

General



We only specified a name in this case for this panel, but it is very useful to also set a link for the panel to tie directly into a report in the webadmin.

2.4 Creating a New Panel (existing data)

What

It is possible in the FileWave dashboard to add panels directly from an inventory query (report) with some limitations (no data aggregation).

When/Why

The data panels are simple enough to add as you'll see below, but the only representation available for them is a table format, which is the equivalent of the report/inventory query itself, but in the dashboard, so easier to share.

How

All Inventory Queries (reports) are available automatically when you choose FileWave Tabular Datasource for your panel's query. See a quick walkthrough below of creating a new table based panel to show all macOS clients:

3.0 Advanced Dashboard Primer

Aren't you an intrepid explorer! Look at you jumping right to the advanced section! Well, we are glad you are here. In this section we'll be looking at how you can create much more complex dashboard content.

As soon as you want to go beyond supplied panels, or direct data from inventory queries, then things are going to get a bit more complex. However, don't be daunted! We are going to give you the building blocks here for building complex and very meaningful aggregate data from FileWave sources.

Please see the articles below for the elements we'll build on:

3.1 Aggregating Data

What

Up until now, we have been talking about the capability of the FileWave dashboard to show data. We have looked at pre-built examples, and done a small amount with data from an inventory query. But the real power in the dashboard is the ability to aggregate (or summarize) data.

When/Why

Let's consider a new deployment of Java to the environment: We can create a report that shows us every client, and every version of Java on those clients, but that doesn't give an easy overview of how the rollout from build 183 to build 196 is going. But, if we could take that same data, and count the number of clients with each version and represent that in a chart, then that would give us the exact picture we might be looking for. This aggregation of data is a very powerful tool.

How

So, we know how to create new dashboards and panels, but how do we actual get aggregated data over to our (grafana) dashboard? The answer to that lies in using Prometheus and configuring a scrape file for collecting and aggregating that data.

3.1.1 Grouping Data Using Prometheus

What

In order to do summary reporting, we need to leverage the power of Prometheus.

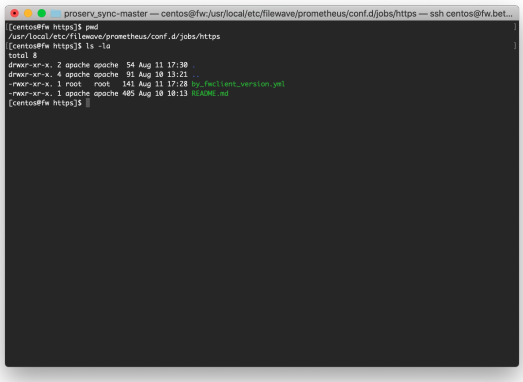
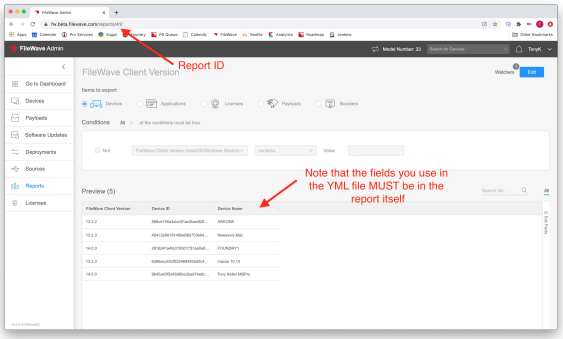
When/Why

Anytime we want to do something like report on a rollout or general status, we are going to want to summarize a report. We will accomplish this by using a Prometheus config file on the FileWave server itself.

How

The configuration (or yml files) that we'll create will always be placed in the /usr/local/etc/filewave/prometheus/conf.d/jobs/https directory on the FileWave server. Anything placed in this directory will automatically be read by Prometheus, and the data presented to our dashboard. (Example yml files can be found in /usr/local/etc/filewave/prometheus/conf.d/jobs)

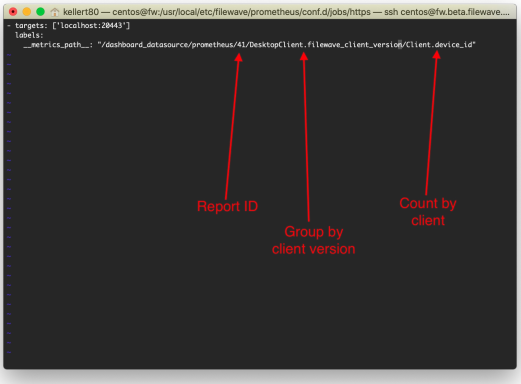
The syntax of these files is quite picky, so it is best to copy an existing one, and then modify it. It may seem complicated, but we are always going to do the following steps:

Step	Example
1. Place a new (or copied) yml file into /usr/local/etc/filewave/prometheus/conf.d/jobs/https with a meaningful name.	
2. Edit the new file to specify the following 3 things: * The inventory query (report) to use * The field you want to count by...device_id is almost always a good one if reporting by device * The field you want summarize (aggregate) by...in this case, the filewave client version	
3. Once your report is created, the report id to use is most easily accessed through the webadmin. Note that the fields you want to use for aggregation must be in the report.	
4. Get the definition for the fields you want to use from the API...the easiest way is to do a curl from the command line like this: `bash curl -s -k -H "Authorization: <Base64 API Token>" https://<my.server.address>:20445/inv/api/v1/query/<report_id> | python -mjson.tool ` Make sure and substitute in your values for the <Base64_API_Token>, <my.server.address> and <report_id> You'll get a response that includes the component and the field names as	

shown at right

```
query/41 |   python -mjson.tool
{
    "criteria": {
        "expressions": [
            {
                "column": "filewave_client_version",
                "component": "DesktopClient",
                "operator": "contains",
                "qualifier": "."
            }
        ],
        "logic": "all"
    },
    "favorite": false,
    "fields": [
        {
            "column": "device_name",
            "component": "Client"
        },
        {
            "column": "filewave_client_version",
            "component": "DesktopClient"
        },
        {
            "column": "device_id",
            "component": "Client"
        }
    ],
    "group": 0,
    "id": 41,
    "main_component": "Client",
    "name": "FileWave Client Version",
    "version": 3
}
```

4. Edit the YML file to specify the 3 items as they match your report definition, then save the file. If using the sample file, remember to take out the comment # at the beginning of each line. Example at right:



Within a minute or two of creation of the file, the data should be available in your dashboard for a new panel.

3.1.2 Testing the Prometheus Scrape

What

Assume for a moment you made a typo in the yml file, or some other problem occurs and your new scrape isn't showing in Grafana...how can you see what is going on?

When/Why

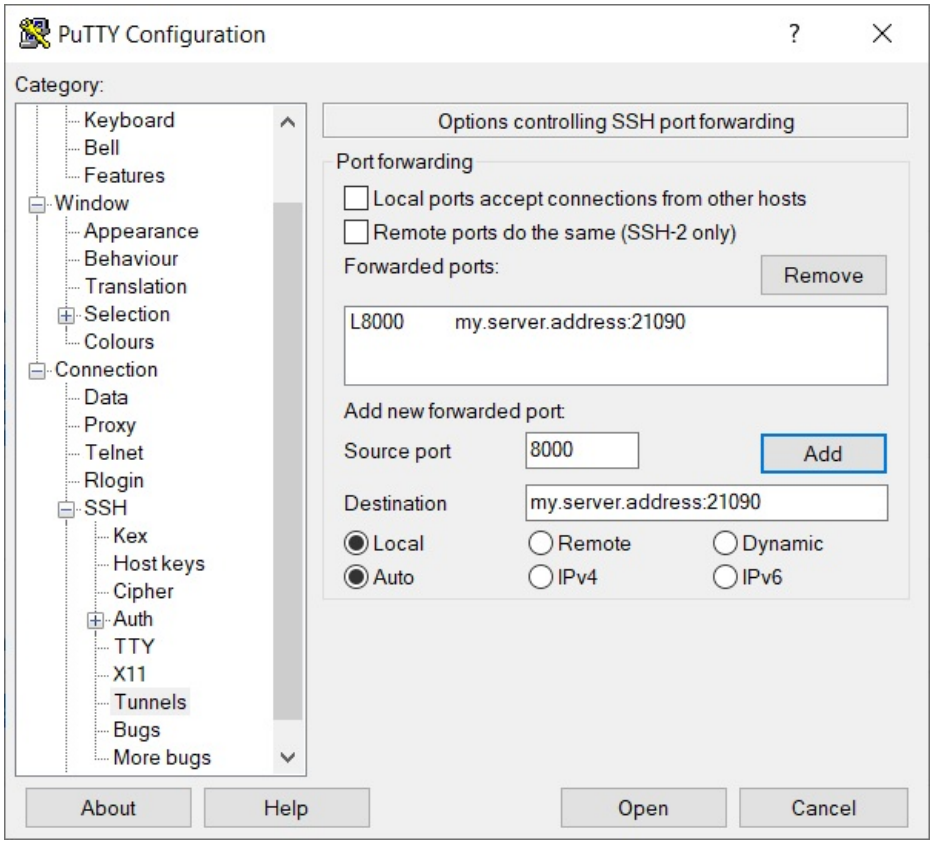
Thankfully there is a service running that allows you to see the status of all Prometheus scrapes, and will usually give you an idea about what is going on. We can check a web page to get this detailed information.

How

The webpage/port in question though is NOT open by default to external systems and we must access it in a special way. If the port were open, we would normally just go to https://my.server.address:21090/targets. But, to work around the port not being opened, we can do the following (from terminal, macOS):

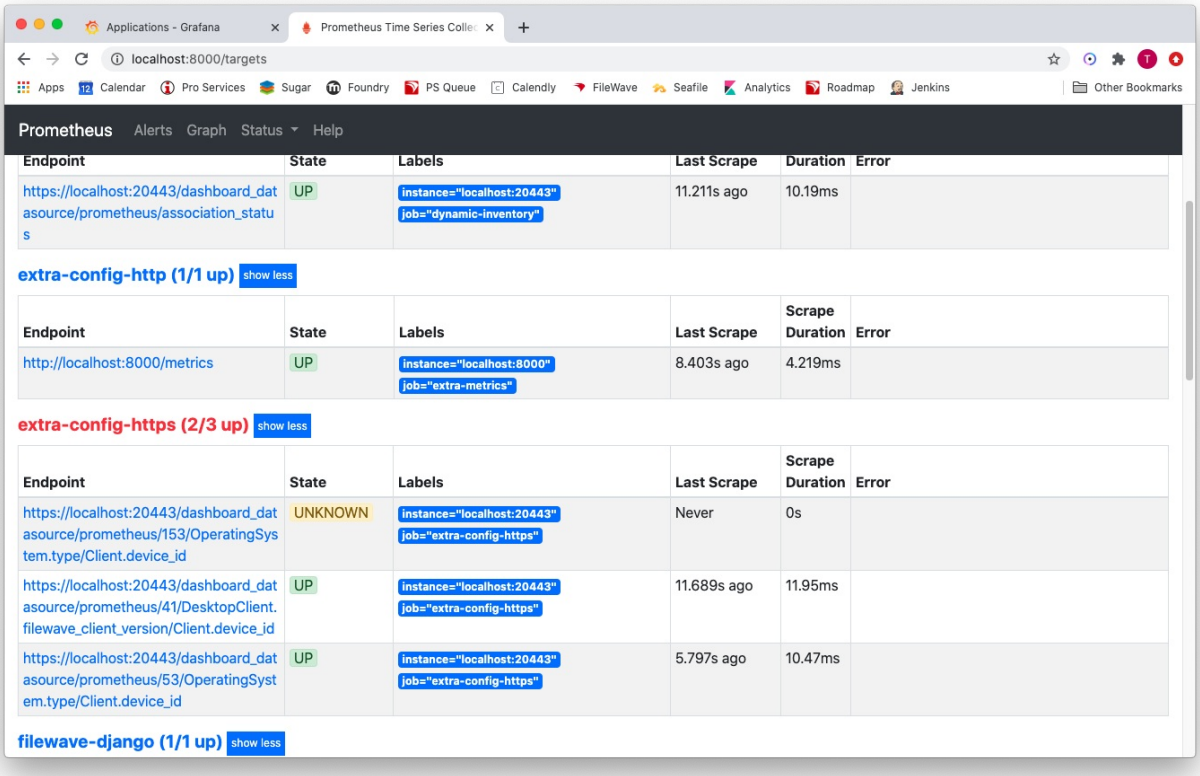
```
ssh -L 8000:localhost:21090 user@my.server.address
```

Alternatively, if you are on a Windows device, you can do the same thing through Putty by configuring a "tunnel" with local port of your choosing redirecting to the FileWave server:



These configurations redirects our requested traffic back to port 8000 on our local device, based on the ssh connection being established. The result is that in our own browser then, we can go to http://localhost:8000/targets to see the scrape data.

See below how I have a mistake in one of my jobs (query 153 doesn't actually exist):


Applications - Grafana
Prometheus Time Series Colle
localhost:8000/targets
Apps
Calendar
Pro Services
Sugar
Foundry
PS Queue
Calendly
FileWave
Seafile
Analytics
Roadmap
Jenkins
Other Bookmarks
Prometheus
Alerts
Graph
Status
Help
Endpoint
State
Labels
Last Scrape
Duration
Error
https://localhost:20443/dashboard_datasource/prometheus/association_status
UP
instance="localhost:20443"
job="dynamic-inventory"
11.211s ago
10.19ms
extra-config-http (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://localhost:8000/metrics
UP
instance="localhost:8000"
job="extra-metrics"
8.403s ago
4.219ms
extra-config-https (2/3 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
https://localhost:20443/dashboard_datasource/prometheus/153/OperatingSystem.type/Client.device_id
UNKNOWN
instance="localhost:20443"
job="extra-config-https"
Never
0s
https://localhost:20443/dashboard_datasource/prometheus/41/DesktopClient.filewave_client_version/Client.device_id
UP
instance="localhost:20443"
job="extra-config-https"
11.689s ago
11.95ms
https://localhost:20443/dashboard_datasource/prometheus/53/OperatingSystem.type/Client.device_id
UP
instance="localhost:20443"
job="extra-config-https"
5.797s ago
10.47ms
filewave-django (1/1 up)
show less

3.1.3 "Exploring" Your New Aggregate Data

What

We don't have to jump right in to creating a proper reporting panel for our new data. Many times simply looking at the data itself can be helpful.

When/Why

It is best to always take a look at your resultant data just to give it a sanity check before you start doing formal reporting. The easiest way to do this is simply to use the "Explore" function within Grafana.

How

The data we previously aggregated from Prometheus will always be presented in the form of filewave_inventory_query_id, where id is the id of the report you based the data upon. You'll see below how we can explore that data:

Notice that the data are no longer 5 individual records, but now 3 records, because we have 3 different versions of the FileWave client between these endpoints. That data is perfect for us to create our very own pie chart.

3.1.4 Creating your Data Panel

What

Now that we have our data being passed over to the dashboard, and we know how to access it, we can build a panel using the visualization that fits our data.

When/Why

Many times, when the data we are presenting is representing the "whole" of an environment, then we might choose to use a pie chart to represent that data, and usually that data is presented in a snapshot form (i.e. only the latest data). At other times, when we want to watch the progression of something, such as number of enrolled devices, we might want to look at that data as it progresses over time in a line graph.

How

Creating this new panel isn't much harder than our previous example, but there are a few new things you'll notice in the overview below:

There are lots of options you can play around with in the different visualizations to customize the output (we won't attempt here to document that). Note though that you can reference data elements for things like the legend. In this case putting {{genericdesktopclient__filewave_client_version}} in as the legend format makes the legend use the much more meaningful field value.

One more note: We chose the "Instant" option in this case because we only want to see the last version of the data for this particular panel (and almost always "instant" for a pie chart). However, the data really is time-series data, so a bar or line chart could show these data elements changing over time to watch progression.

3.2 Extra Metrics

What

We learned in the 3.1 section how to build our own custom panels. "Extra Metrics" is an independently built tool to automate creation of a few reporting elements for us without doing it manually. This solution is NOT directly supported by FileWave, but you may find it useful in your environment.

When/Why

"Extra Metrics" gives data on applications, and generally on patch status of your devices. The patch status elements are hard-coded, but the application versions panels are driven by dynamic reports that you can tweak to fit your needs. All information for installation and upgrade of this solution is found here: https://pypi.org/project/filewave-extra-metrics/#description

How

In this example, we have Extra Metrics installed, and are adding a new report to view Firefox information on macOS devices.

4.0 Dashboard Alerts

With the Custom Dashboards configured you can now choose to configure email alerts based upon conditions met. Alerts are only available for Visualisations of type Graph.

Custom Dashboards provide live information, but for some you will want to receive notifications when certain criteria has been reached. Two elements require configuring, email and the alert itself: Email and Alert

4.1 Grafana Email Configuration

What

Use the below method to configure email in Grafana

When/Why

Email must be configured in advance of creating any Alerts.

How

At this time, it involves editing a Grafana ini file.

```
/usr/local/etc/filewave/grafana/conf/filewave.ini
```

Add the following smtp section, edited to match your credentials, into the above filewave.ini file:

```
#################################### SMTP / Emailing ##########################
[smtp]
enabled = false
host = localhost:25
user =
# If the password contains # or ; you have to wrap it with triple quotes. Ex """#password;"""
password =
cert_file =
key_file =
skip_verify = false
from_address = admin@grafana.localhost
from_name = Grafana
ehlo_identity =

[emails]
welcome_email_on_sign_up = false
templates_pattern = emails/*.html
```

Details on configuring the filewave.ini file may be found at Grafana's Configuration Documentation:

> Do not edit entries already created within this file.

Once done, restart the FileWave Server Service:

```
filewave server restart
```

From the Web Admin, choose

- Dashboard > Alerting (Bell Icon) > Notification Channels

select New Channel and add details as desired. Once done choose Send Test. You should receive an email.

4.2 Grafana Alert Configuration

What

Set up Alerts for those times you want to be notified.

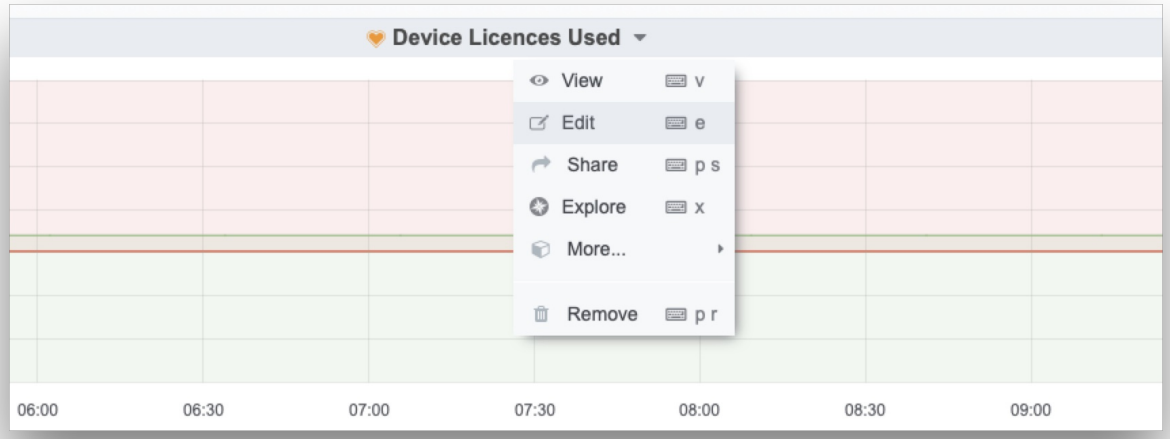
> Alerts are only available for Visualisations of type Graph

When/Why

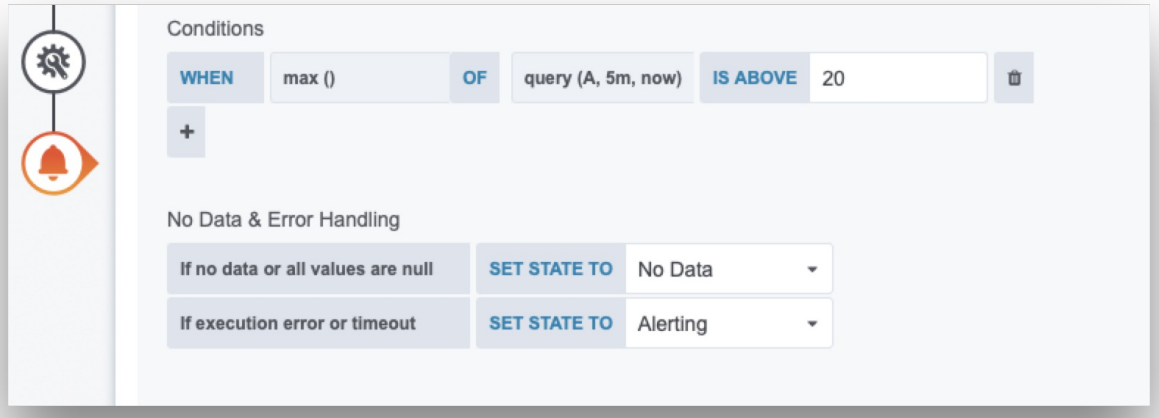
Leverage your Web Admin Custom Dashboard elements to build Alerts.

How

Select your chosen Graph and from the drop down choose 'Edit':



Select the Notification icon and configure the conditions as desired. In this example if the total device count exceeds 20 devices, a notification will be triggered.



Add any message as you see fit.

5.0 Grafana Plugins

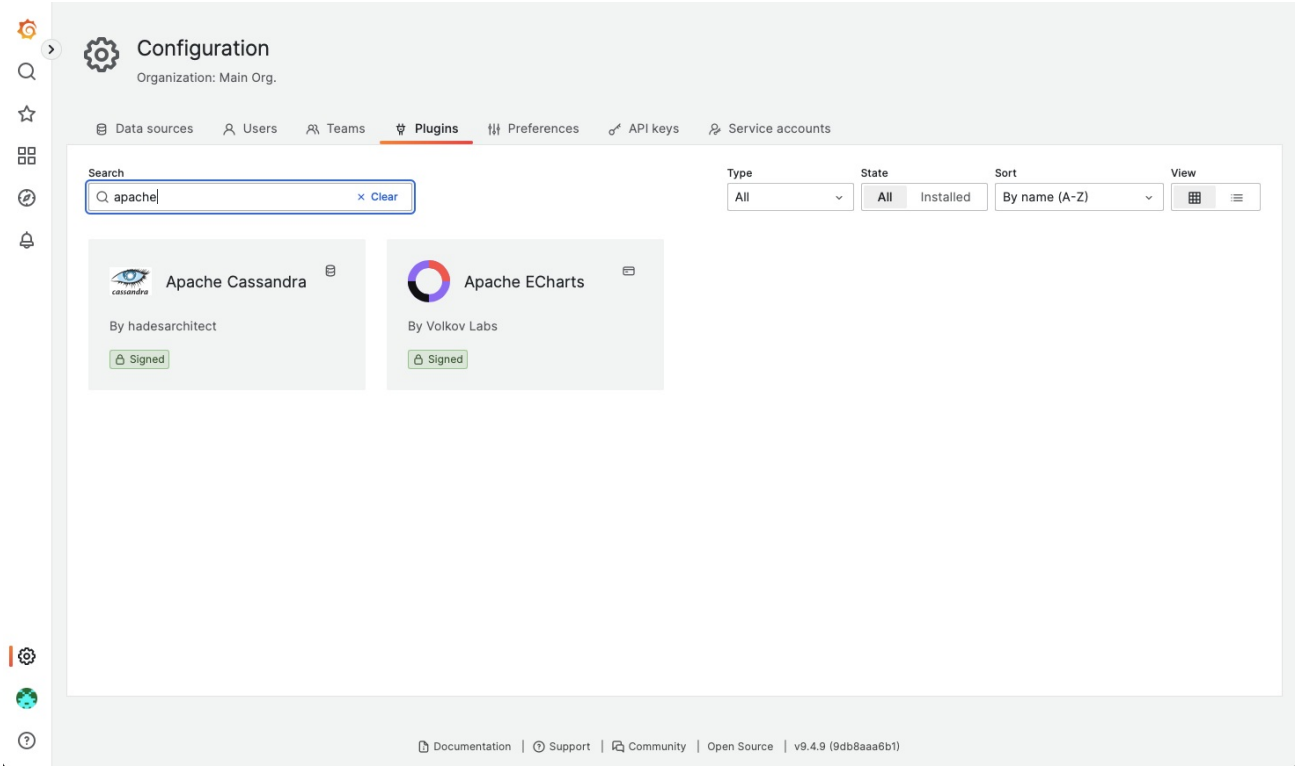
Plugin Installation

Plugins are a great way to extend the features of Grafana, optimising the Grafana experience.

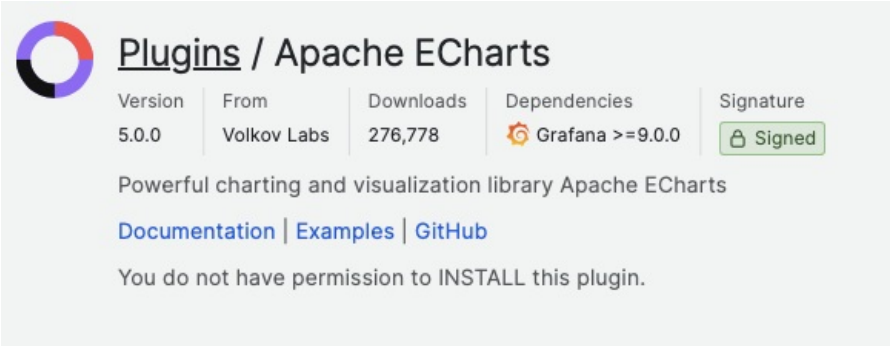
> Recommendation is that any installed plugins are signed. All plugins from the built-in search should be approved already, but plugins are available from other sources.

Installation

Plugins are easily searchable through the Grafana Dashboard, via the Configuration tabs



On selecting a plugin though, at the bottom of the description it will indicate necessary permissions are not enabled:



Installation through the Grafana GUI requires a Grafana Admin. FileWave does not allow installation of plugins in this manner. Instead, the command line should be used directly on the server to instal plugins.

Command Line Installation

Scrolling down through the plugin page, there is a section on using command line to instal the plugin, however, this should not be used. It does however highlight the required plugin name to be used in the process.

Taking Apace Charts as an example:

Getting Started

Apache ECharts visualization panel can be installed from the Grafana Catalog or utilizing the Grafana command line tool.

For the latter, use the following command.

```
grafana-cli plugins install volkovlabs-echarts-panel
```

From the description, the plugin name is:

`volkovlabs-echarts-panel`

As such the command to instal this plugin on a FileWave Server would be:

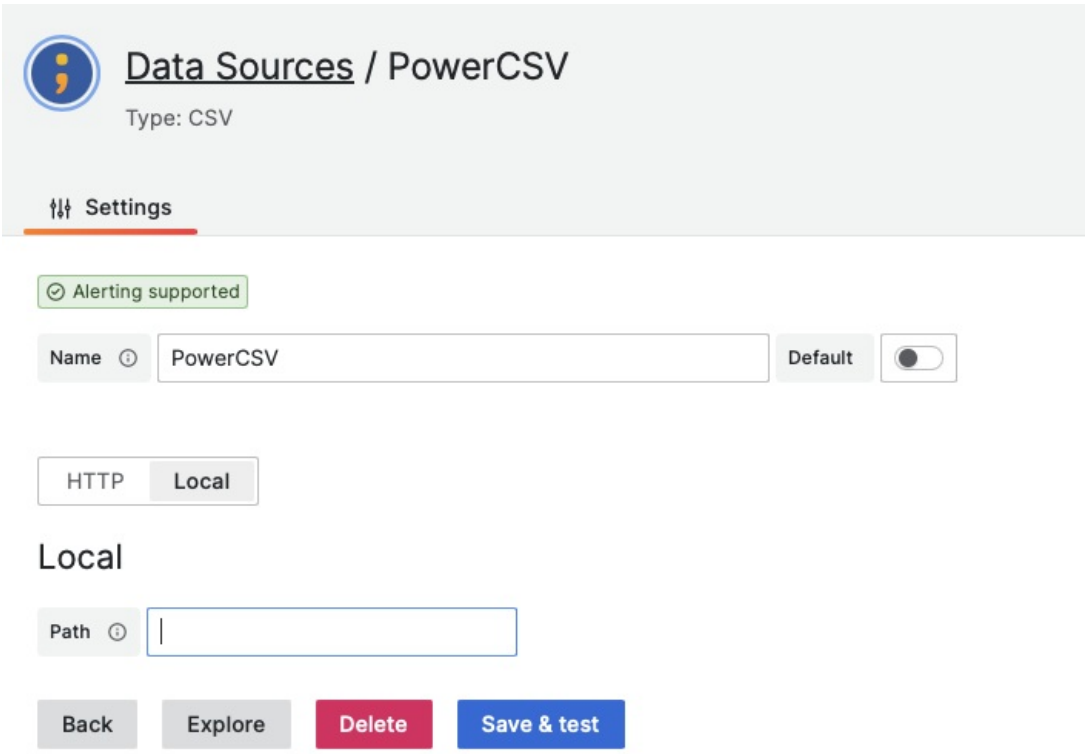
```
/usr/local/sbin/grafana-cli --pluginsDir=/usr/local/filewave/instrumentation_data/grafana/plugins --
homepath=/usr/local/filewave/grafana plugins install volkovlabs-echarts-panel
```

The process should report success and the plugin should now be available.

Plugin Configuration

Some plugins may require additional configuration, e.g. they require reference to additional files. An example of this is the CSV plugin by Marcus Olsson.

When selecting the CSV Plugin as a Data Source, the path to the CSV file is required, with a choice of HTTP or Local:



Local Data Source

By default, local data sources are not allowed. It is possible to overcome this via the Grafana 'ini' file, however this has limitations.

The FileWave Grafana 'ini' file is defined as:

`/usr/local/etc/filewave/grafana/conf/filewave.ini`

Enablement of Local files is achieved by adding the following into this file:

```
[plugin.marcusolsson-csv-datasource]
allow_local_mode = true
```

After doing so, the server service should be restarted:

```
fwcontrol server restart
```

Local files may now be accessed with this plugin. However, the caveat is that this file is overwritten by the FileWave installer on each upgrade. The outcome is the file must therefore be edited on each upgrade attempt.

HTTP Data Source

HTTP Data Source has no similar caveat. If there is already an available HTTP share in-house, this could be utilised. However, the FileWave Server could have the apache configuration adapted to include such files.

Configure Apache

This process will rely on Apache having access to a directory which serves these files. This could either be one that already exists or configuration could be used to add a new directory to serve just the required files.

Recommendation would be to create a new directory.

This is editing FileWave Server files and folders. FileWave upgrades can overwrite files, folders or expansion of product may include new files and folders. Consider using a directory that is unlikely to clash with future updates.

In this example, the new chosen folder is:

`/usr/local/filewave/grafana_plugins_custom/csv`

'grafana_plugins_custom' does not yet exist. Rather than just creating one custom plugin folder for all possible plugin use, in this example a subdirectory for the CSV files has been created. Additional plugins that also required http file shares could use the same parent, but have their own subfolder.

The Apache file that can be edited to create new http file shares is:

`/usr/local/filewave/apache/conf/httpd_custom.conf`

As a FileWave Custom file, this will persist with Filewave upgrades.

```
Alias /plugins_custom /usr/local/filewave/grafana_plugins_custom/csv
<Directory "/usr/local/filewave/grafana_plugins_custom/csv">
Options Indexes FollowSymLinks
AllowOverride All
Order allow,deny
Allow from all
</Directory>
```

Full details on each option can be found in the Apache Documentation

However, to highlight two key aspects.

Directory

This is the newly created directory from which the files may be shared from

Alias

When accessing the files from a URL, this is essentially a shortcut for the URL, which will be redirected internally to this folder on access. In this instance, the alias (shortcut link) is:

`/plugins_custom`

Other options should be configured as required.

FileWave Server process should be restarted once configured:

```
fwcontrol server restart
```

Downloading CSV files

Once configured, CSV files may be populated into the newly created document. For example:

```
# ls -l /usr/local/filewave/grafana_plugins_custom/csv
total 184
-rw-r--r--  1 root  _www  14425 Aug  3 17:06 admin.csv
-rw-r--r--  1 root  _www  19364 Aug  3 17:06 data.csv
-rw-r--r--  1 root  _www  36694 Aug  3 17:06 paint.csv
-rw-r--r--  1 root  _www  19345 Aug  3 17:06 video.csv
```

Test download may be achieved by way of the server URL appended with the alias. For example if the server were called 'demo.filewave.ch', then the URL from the above configuration would be:

`https://demo.filewave.ch/plugins_custom/`

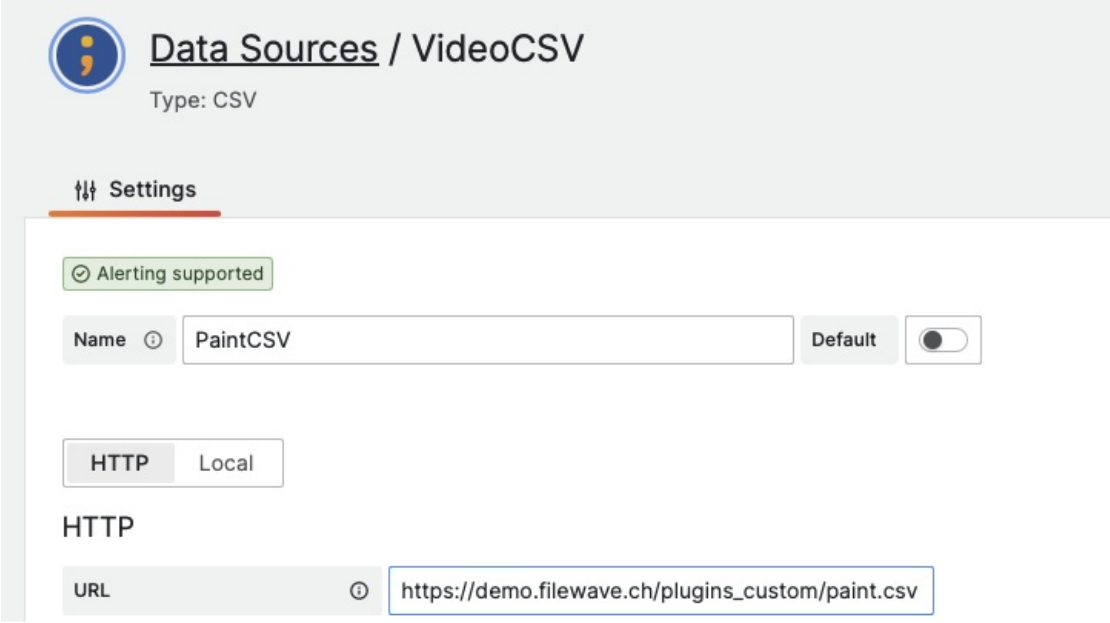
However, a file should be addressed directly. Taking the paint.csv as an example:

`https://demo.filewave.ch/plugins_custom/paint.csv`

Accessing this URL in a browser should show the contents or download the file.

Plugin Data Source

Now an Apache share has been created with populated files, a Data Source may be configured to use HTTP for access to the files.



Other Data Source settings should be configured as desired.

Troubleshooting

Custom Grafana Dashboard - YML Files not being processed

What

You may notice that your Grafana dashboard won't process data from YML files. This can happen if the API key is regenerated for FileWave, but the file that Grafana uses is not updated. As recently as FileWave 14.9.0, this can be an issue with an easy solution outlined below. Development is looking at possibly always overwriting this file at server startup which would eliminate this problem from appearing, and it should only appear as an issue if the API key for the server is regenerated, causing it to not match. For most customers, this is not likely to be an issue.

When/Why

When you 3.1.2 Testing the Prometheus Scrape as you set up a custom Dashboard with a YML file, you may see that the files show "DOWN," as seen below. This is a sign that you are experiencing this issue. You'll also see no data on your widgets in the Dashboard.

dynamic-inventory (0/1 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
https://localhost:20443/dashboard_datasource/prometheus/association_status	DOWN	instance="localhost:20443" job="dynamic-inventory"	35.28s ago	37.125ms	server returned HTTP status 401 Unauthorized

extra-config-https (0/7 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
https://localhost:20443/dashboard_datasource/prometheus/298/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	34.858s ago	173.401ms	server returned HTTP status 401 Unauthorized
https://localhost:20443/dashboard_datasource/prometheus/296/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	56.275s ago	240.322ms	server returned HTTP status 401 Unauthorized
https://localhost:20443/dashboard_datasource/prometheus/299/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	1m 58s ago	318.845ms	server returned HTTP status 401 Unauthorized
https://localhost:20443/dashboard_datasource/prometheus/301/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	2m 7s ago	96.661ms	server returned HTTP status 401 Unauthorized
https://localhost:20443/dashboard_datasource/prometheus/300/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	2m 28s ago	24.173ms	server returned HTTP status 401 Unauthorized
https://localhost:20443/dashboard_datasource/prometheus/295/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	2m 38s ago	28.537ms	server returned HTTP status 401 Unauthorized
https://localhost:20443/dashboard_datasource/prometheus/297/Client.serial_number/Client.device_id	DOWN	instance="localhost:20443" job="extra-config-https"	1m 12s ago	53.187ms	server returned HTTP status 401 Unauthorized

How

To fix this issue, SSH to your server and remove the file, as shown below. the bearer_token_file has the wrong API key, and if you restart FileWave after deleting it, then a new correct one will be regenerated. If you are a hosted customer, then you will need support to remove this file for you, and very likely, they helped you upload the YML file, to begin with.

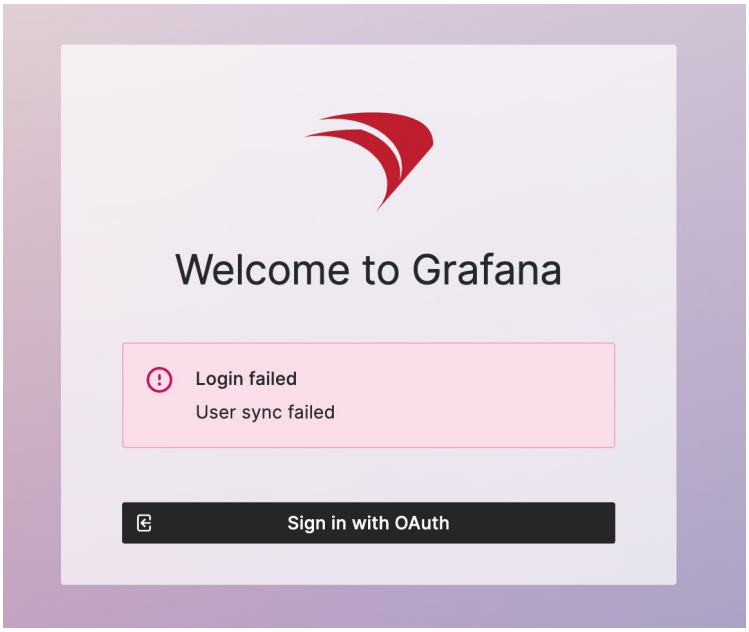
```
rm /usr/local/etc/filewave/prometheus/conf.d/bearer_token_file
/usr/local/bin/fwcontrol server restart
```

Troubleshooting

Resolving “Login failed: User sync failed” Error in Grafana

What

This article addresses the issue of encountering a “Login failed: User sync failed” error when logging into Grafana through FileWave Anywhere.



When/Why

This error occurs when there is a username conflict between Grafana and FileWave. Specifically, if an administrator user named “admin” is created in FileWave Central, it conflicts with Grafana’s internal use of the “admin” username. This issue is common when administrators use the “admin” username in FileWave Central -> Preferences -> Manage Administrators and then attempt to log in to the FileWave Dashboard via Grafana.

How

To resolve this issue, you need to avoid using the “admin” username in FileWave. Follow these steps assuming the user is a built-in user rather than one coming from an LDAP or IdP:

1. Access FileWave Central:
   - Open FileWave Central.
   - Navigate to Preferences -> Manage Administrators.
2. Rename the “admin” Account:
   - Identify if an “admin” user exists.
   - Rename this user to a specific named account, such as “Bob Randall”.
   - Ensure the new username is unique and does not conflict with other system usernames.
3. Create Named Accounts:
   - Create individual named accounts for each administrator. For example, use names like “Jane Doe” or “John Smith”.
   - This practice not only prevents conflicts with Grafana but also ensures clarity and accountability.
4. Update Login Credentials:
   - Inform all administrators about their new login credentials.
   - Update any saved login credentials in browsers or password managers accordingly.

Related Links

- Managing FileWave Administrators

Digging Deeper

Grafana, a popular open-source platform for monitoring and observability, utilizes a default “admin” user account for its initial setup. FileWave administrators often use the “admin” username for simplicity, which leads to conflicts. It is a best practice to avoid generic usernames such as “admin” or “root” in any system to prevent potential conflicts and security issues.

By adopting named accounts, organizations can enhance their security posture and improve user accountability. Named accounts make it easier to track who made specific changes, which is crucial for auditing and compliance purposes.

For more information on managing administrators in FileWave and avoiding username conflicts, refer to the official FileWave and Grafana documentation linked above.