

How To Configure Filters In Nagios Log Server 2024R2

Purpose

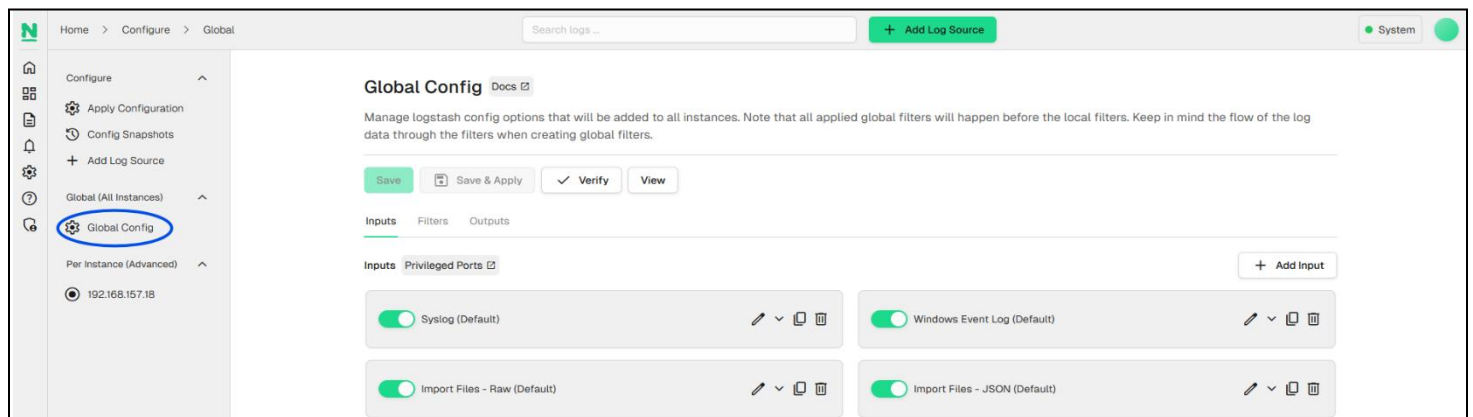
This document describes how to customize Nagios Log Server Filters, it is intended for use by Nagios Log Server 2024R2 Administrators.

What Are Filters?

Filters can be applied to messages before they are sent to Opensearch for indexing. They perform actions such as breaking apart messages into fields for easy searching, adding geo location information, resolving IP to DNS names and dropping messages you do not want indexed.

Filter Configuration Location

Nagios Log Server is a cluster-oriented application that uses Logstash to receive and process logs. The base configuration provided with a default installation of Nagios Log Server has all the filters defined as part of the Global Config. Global Config is an easy way to set up the same Logstash configuration on all your instances. To access the configuration, navigate to **Configure > Global (All Instances) > Global Config**.



On the **Global Config** page there are two main tables named **Inputs** and **Filters**, and a third table called **Outputs**. Inputs, Filters and Outputs are all used by Logstash to process incoming log data and do something with it, which normally is to store it in the Opensearch database. This document will focus on Filters.

How To Configure Filters In Nagios Log Server 2024R2

Filters Explained

The filters are in a structured format like this:

```
<filter_type> {  
    <filter_action> => [ '<matching_field>', '<matching_pattern>' ]  
}
```

<filter_type> is the filter plugin that will be used for the <filter_action>. Logstash allows a large amount of possible filter types, this documentation will explore some that are useful for manipulating logs.

This example will explain how the grok <filter_type> can be used for filtering. Grok is a plugin that is used by Logstash for making specific filters using regular expressions and matching. The grok documentation explains it as:

"Grok is currently the best way in logstash to parse crappy unstructured log data into something structured and queryable".

The grok plugin allows a lot of customization and will be used heavily in making custom filters in your Nagios Log Server configuration:

<filter_action> is the action that will be taken using the filter type. Common grok actions are match, add_field and add_tag.

<matching_field> will be the log field that is being processed. When Logstash receives logs from sources like syslog, the received data is categorized by certain fields (like the message field that contains some text about the log entry). You can choose things like hostname, severity, message and many more.

<matching_pattern> relates to the pattern that will be looked for in the <matching_field>, the data matched by the pattern(s) can be turned into new fields stored in the Opensearch database.

A real-world example will help explain this in more detail. In this example the field message will be the <matching_field>. Here is an example of a line from an Apache error_log:

How To Configure Filters In Nagios Log Server 2024R2

```
[Wed Oct 22 12:45:58 2014] [error] [client 192.168.5.19] File does not  
exist: /usr/local/nagiosxi/html/someURL
```

This message from the Apache error_log shows that the someURL file does not exist. It shows the time the notice was posted, the type [error], the client trying to access the someURL file, the log message and the path that generated the request error.

What is important is that the data on this line follows a structured format, every time an entry is received from this input the structure of the entry will always be the same. Identifying this structure is done by using the `<matching_pattern>` to break this string of text into fields. Here's the structured data broken down:

```
[Wed Oct 22 12:45:58 2014]  
[error]  
[client 192.168.5.19]  
File does not exist: /usr/local/nagiosxi/html/someURL
```

To take control of this message and filter the whole message or parts of it you will use a grok pattern that relates to the pattern of the message.

The filter shown below will be applied to any logs with the apache_error program name. This is one of the two default filters that are present inside Nagios Log Server.

```
if [process][name] == 'apache_error' {  
    grok {  
        match => [ 'message', '%{HTTPD_ERRORLOG}' ]  
    }  
    mutate {  
        replace => [ 'type', 'apache_error' ]  
    }  
}
```

At this point the focus will be on the match line. The first line is explained in the [Filter Conditions](#) section and the mutate lines will be explained in the [Mutate](#) section.

How To Configure Filters In Nagios Log Server 2024R2

Match

The line starts off like this:

```
match => [ 'message' , '
```

The 'message' is the `<matching_field>` field that contains the data you want to break into fields, take note that it is surrounded by single quotes. You can also see that the square bracket `[` is used to begin the match definition (filter structure). A single quote is then used to start the `<matching_pattern>`.

The `<matching_pattern>` here handles matching the message component of the log against the "HTTPD_ERRORLOG" pattern.

For more information about Grok usage, please visit OpenSearch's documentation [here](#)

Mutate

Mutate is another very useful plugin used in Logstash filtering. This allows you to replace or append a value in any field with a new value that may replace the whole field or add, delete or append portions of it. Here is the mutate line from earlier:

```
mutate {  
    replace => [ 'type', 'apache_error' ]  
}
```

In our example above we are changing the log type which would be syslog and replace it with `apache_error`. This allows you to differentiate between normal syslogs and apache syslogs.

How To Configure Filters In Nagios Log Server 2024R2

Filter Conditions

A filter can be restricted to certain logs by using a simple if statement:

```
if [program] == 'apache_error' {  
    grok {  
    }  
    mutate {  
    }  
}
```

Here you can see that the [program] must be apache_error for these filters to be applied.

How does the log received by Logstash know that the program is apache_error? When you run the setup script on your Linux server that is running Apache it will define the name as apache_error.

In Nagios Log Server navigate to **Configure > Add Log Source** and select Apache Server.

Under the Run the Script heading you can see the following line of code:

```
sudo bash setup-linux.sh -s nagios_log_server -p 5544 -f "/var/-  
log/httpd/error_log" -t apache_error
```

You can see that apache_error is being defined as part of the setup script. The syslog application on the sending server will define these log entries as coming from apache_error. This completes the explanation of the above example for Apache error_log messages.

How To Configure Filters In Nagios Log Server 2024R2

Patterns

Patterns take complex regular expressions and define them as understandable words. For example:

```
MONTHDAY = (?:0[1-9])|(?:[12][0-9])|(?:3[01])|1-9
GREEDYDATA = .*
WORD = \b\w+\b
```

As you can see, the defined patterns are complicated and difficult to refer to. Words like “MONTHDAY” are useful shorthand names for these useful patterns.

If you would like to see a list of available patterns you will need to refer to the source code.

Here is the source code for grok patterns:

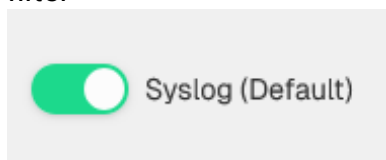
<https://github.com/hpcugent/logstash-patterns/blob/master/files/grok-patterns.json>

All available patterns are in different files which are located here:

<https://github.com/logstash-plugins/logstash-patterns-core/tree/master/patterns>

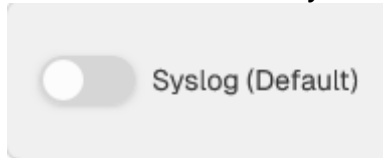
Filter Configuration Options

1. Navigate to **Configure > Global (All Instances) > Global Config**. In the Filters table there are several pre-configured filters that come as part of Nagios Log Server, these are called blocks. The blocks have several icons which are explained as follows.
2. Each of these blocks are Active by default, which is indicated by the Active box next to each filter



How To Configure Filters In Nagios Log Server 2024R2

3. When you click the Active box, the filter will be marked as Inactive and the next time the configuration is applied this filter will not be part of the running configuration. This allows you to save filters even if you don't want to use them right away



4. Clicking the box again will toggle it back to Active



Allows you to rename the block



This will expand the block to show the filter configuration in a text field

Please refer to the [Filters Explained](#) section of this document for more information

The icon will flip upside-down when open, clicking the icon again will collapse the block



This will create a copy of that block

Allows you to easily create a new filter based off the configuration of an existing filter



Delete a block when it is no longer required

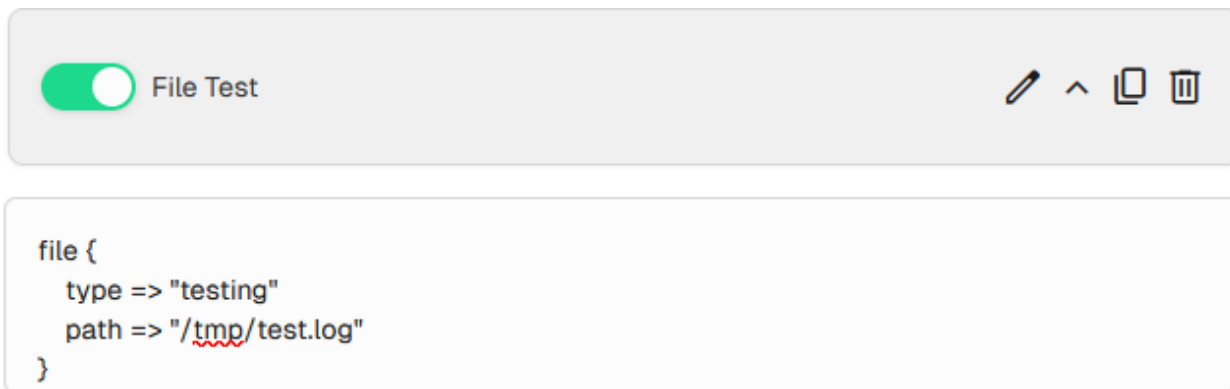
Any changes you make will not be saved until you click the **Save & Apply** button. Keep this in mind as when you navigate away from the page, unsaved changes will be lost.

How To Configure Filters In Nagios Log Server 2024R2

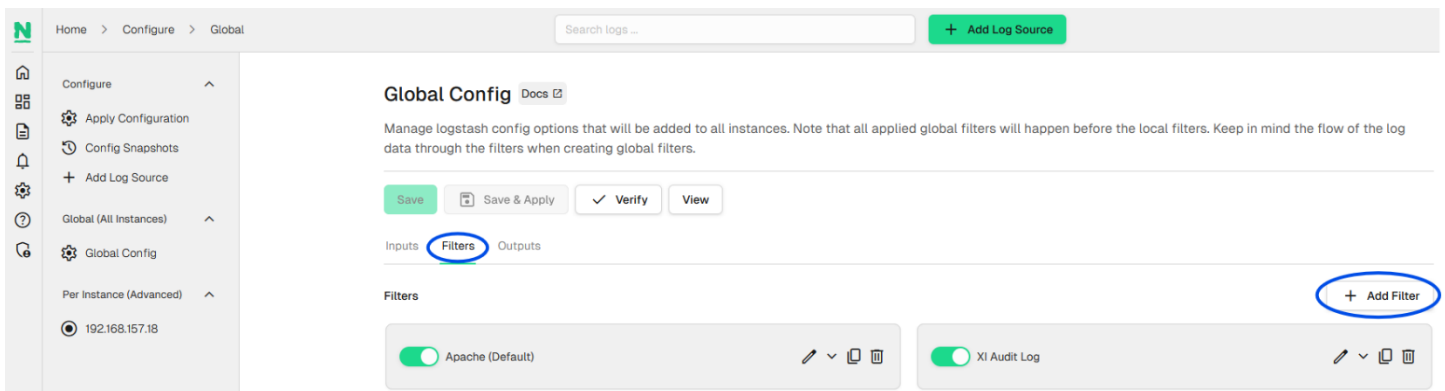
Adding A Filter

The best way to learn how filters work is to add one and then send some log data. This could become a very complicated topic however the following example is kept simple.

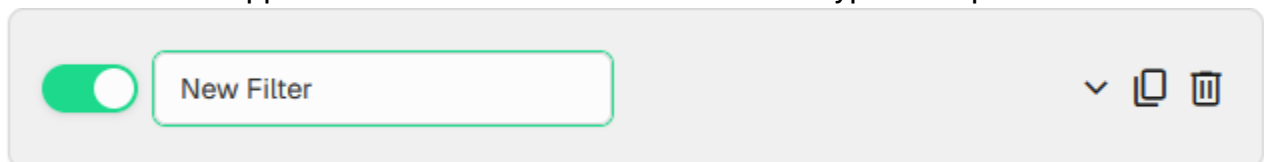
The following example is going to rely on the File Test Input that was demonstrated in the [Configuring Nagios Log Server Inputs](#) documentation. Please return here after you have the input created, here is an example screenshot of the input:



1. On the Global Config page click the **Add Filter** button.



2. A new filter will appear at the bottom of the list of Filters. Type a unique name for the filter.

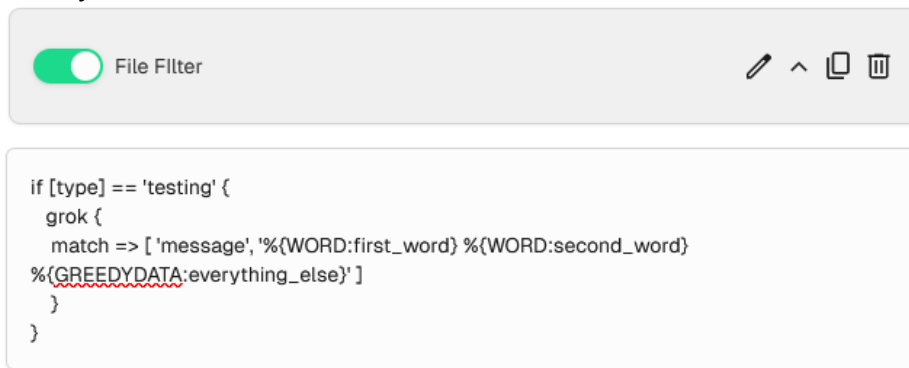


How To Configure Filters In Nagios Log Server 2024R2

3. In the text field you will need to define the filter configuration. Paste the following example:

```
if [type] == 'testing' {  
    grok {  
        match => [ 'message', '%{WORD:first_word} %{WORD:second_word}  
%{GREEDYDATA:everything_else}' ]  
    }  
}
```

Once you have finished click the **Save** button.



The screenshot shows a user interface for configuring a 'File Filter'. At the top, there is a toggle switch labeled 'File Filter' which is currently turned on (green). To the right of the toggle are icons for editing (pencil), undo (up arrow), copy (document), and delete (trash). Below the toggle is a text area containing the following configuration code:

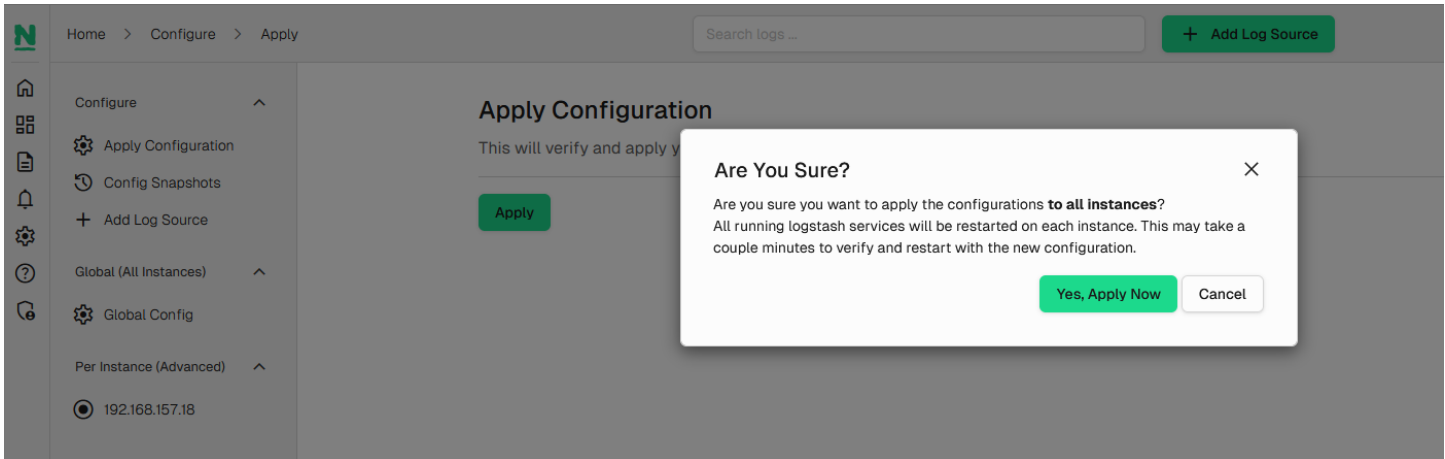
```
if [type] == 'testing' {  
    grok {  
        match => [ 'message', '%{WORD:first_word} %{WORD:second_word}  
%{GREEDYDATA:everything_else}' ]  
    }  
}
```

For the new filter to become active you will need to [Apply Configuration](#), however it is recommended that you [Verify](#) the configuration first. The next step will be to [Apply Configuration](#) to put this new filter into the running configuration so it can be tested and demonstrated.

Apply Configuration

To apply the configuration, in the left-hand pane under **Configure** click **Apply Configuration**. Click the **Apply** button and then on the modal that appears click the **Yes, Apply Now** button.

How To Configure Filters In Nagios Log Server 2024R2



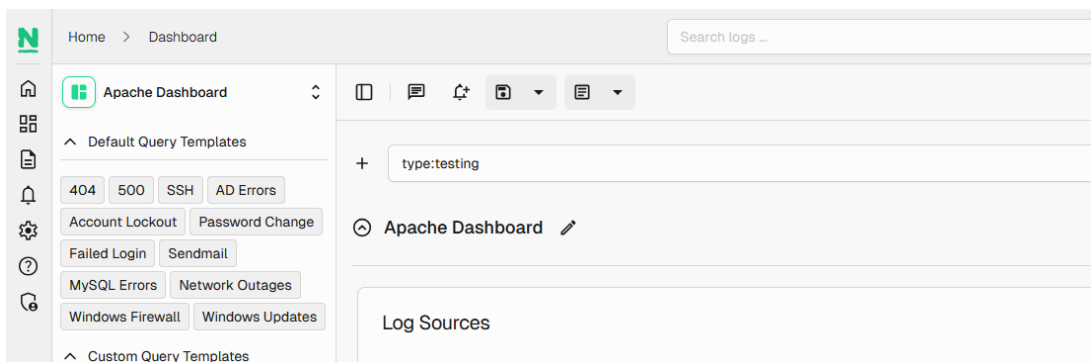
Wait while the configuration is applied. When it completes the screen will refresh with a completed message. Please proceed to the [Test Filter](#) section.

Test Filter

1. Using the filter example created previously, these steps will show you how to confirm the filter is working. Establish a terminal session with your Nagios Log Server instance and then execute the following command:

```
echo "This is a test log entry" >> /tmp/test.log
```

2. Now in Nagios Log Server open the Dashboards page, Advanced Search, and perform the query `type:testing` as per this screenshot:



How To Configure Filters In Nagios Log Server 2024R2

The screenshot displays the Nagios Log Server 2024R2 interface. On the left, the 'All Events' panel shows a list of log entries. One entry is highlighted with a blue box: '2024-12-10T21:41:30.099564302Z'. On the right, the 'Log Entry Details' panel shows the full details of the selected log entry. The details are organized into a table with 'Key' and 'Value' columns. The 'message' field is highlighted with a blue box, and its value is 'This is a test log entry'. The 'first_word' field is also highlighted with a blue box, and its value is 'This'. The 'second_word' field is highlighted with a blue box, and its value is 'is'. The 'everything_else' field is highlighted with a blue box, and its value is 'a test log entry'.

Key	Value
_index	"logstash-2024.12.10"
_id	"Qn2FspMBed8B5lpk.JG9y"
_score	8.425357
@version	"1"
log_path	"/tmp/test.log"
second_word	"is"
original	"This is a test log entry"
@timestamp	"2024-12-10T21:41:30.099564302Z"
type	"testing"
message	"This is a test log entry"
first_word	"This"
everything_else	"a test log entry"

The query should return one result in the **ALL EVENTS** panel.

3. Clicking on the log entry will show you the full details about the entry.
4. Here you can see that the filter successfully broke the message field into the three fields of `first_word` (this), `second_word` (is) and `everything_else` (a test log entry).

So far you can see it is relatively simple to create a filter to break apart the log data into separate fields. Earlier on it was explained that it is very important that the overall pattern you are defining is accurate, otherwise the pattern will not be matched, and the log data will not be broken into fields in the OpenSearch database.

So how do you know the pattern isn't matching and something is wrong? In your terminal session execute the following command to submit another log entry (notice how a colon `:` was added):

```
echo "This: is a test log entry" >> /tmp/test.log
```

How To Configure Filters In Nagios Log Server 2024R2

Log Entry Details

Search elements...

EntryJSON

Key	value
_index	"logstash-2024.12.10"
_id	"gn2LspMBed8B5lpkmnHK"
_score	8.111745
@version	"1"
log_path	"/tmp/test.log"
second_word	"a"
original	"This: is a test log entry"
@timestamp	"2024-12-10T21:48:33.720654755Z"
type	"testing"
message	"This: is a test log entry"
first_word	"is"
everything_else	"test log entry"
name	"localhost.localdomain"

Now in Nagios Log Server on the Dashboards page perform the query `type:testing` and have a look at the new log entry. Here you can see that the filter isn't working as expected, it broke the message field into the three fields of `first_word` (is), `second_word` (a) and `everything_else` (test log entry).

How To Configure Filters In Nagios Log Server 2024R2

What is happening is that the colon that was part of the first word "This:" is causing the grok pattern to start at the "is" word. The grok pattern does not have a colon in it, and lacks pattern syntax that matches it. Sometimes complicated grok patterns will simply add a tag field with the value of `_grokparsefailure` when it does not match. When troubleshooting grok patterns, try and simplify it, remove a section and see if it works. If it does, then the problem was likely in the section you removed.

The remainder of this documentation explains the buttons available on the Global Config page.

Verify

Global Config [Docs](#)

Manage logstash config options that will be added to all instance data through the filters when creating global filters.

Verify the syntax of this portion (global) of the config file.

Save



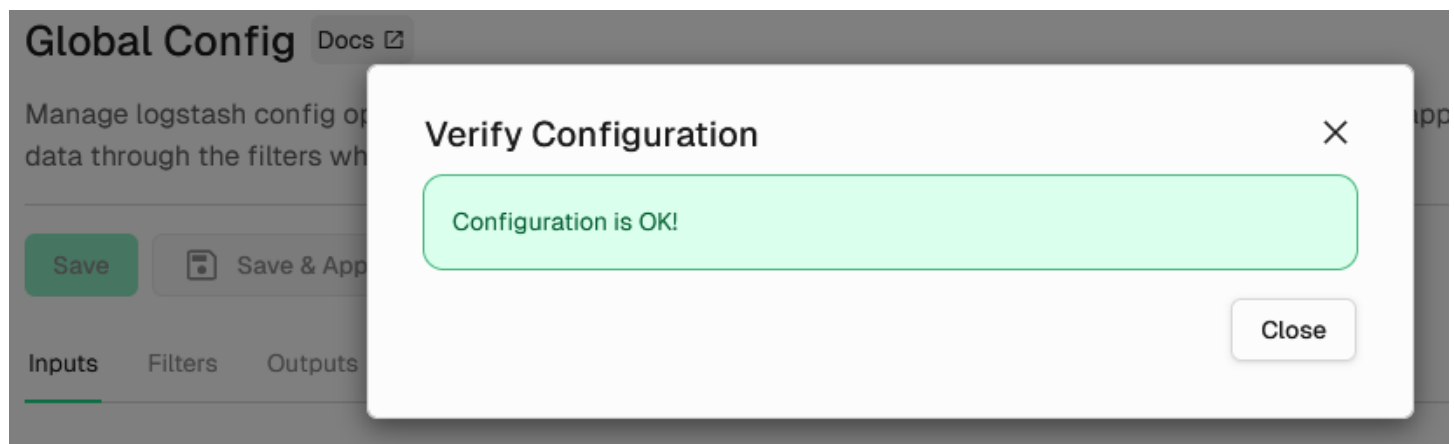
Save & Apply



Verify

View

The **Verify** button ensures that the current saved configuration is valid. It can be useful when updating your configurations before attempting to Apply Configuration.

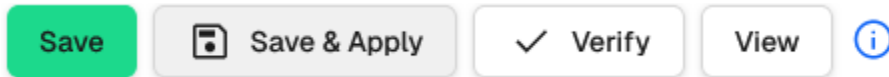


Wait while the configuration is verified.

If you do not receive a "Configuration is OK!" message, then you will need to fix the problem before applying the configuration.

How To Configure Filters In Nagios Log Server 2024R2

Save vs Save & Apply

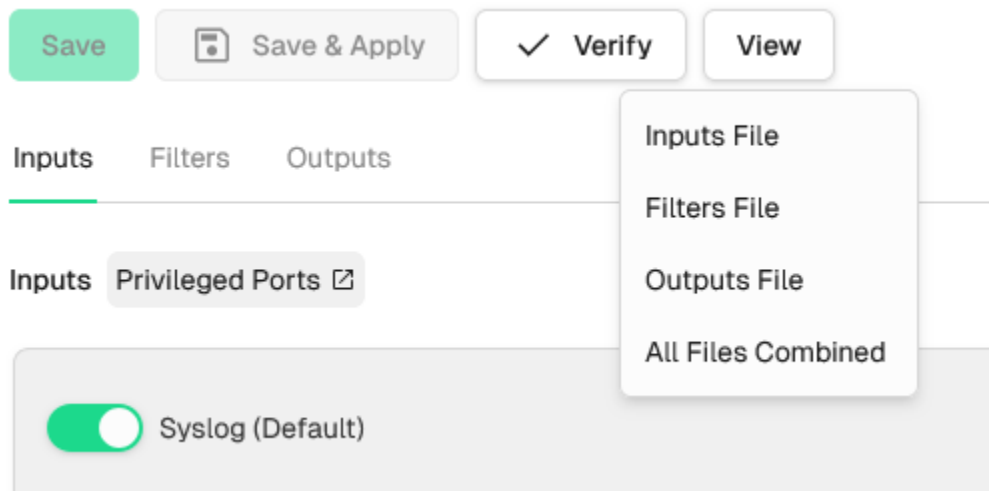


There are two separate buttons, **Save** and **Save & Apply**.

Save allows you to make changes but not make the changes become immediately active. This allows you to navigate away from the Configure screen without losing your work.

Save & Apply will save your changes and immediately apply the configuration changes. This can be helpful if you changed a simple option in your config that does not need to be verified.

View



The **View** button allows you to view the raw Logstash configuration. When you click the button, you have a choice of selecting an individual config or a combination of all the configs.

This will open a new modal window where the configuration can be viewed or copied.

How To Configure Filters In Nagios Log Server 2024R2

External Resources

Advanced Logstash Configuration documentation:

<https://opensearch.org/docs/latest/tools/logstash/advanced-config/>

Here is the source code for grok:

<https://github.com/logstash-plugins/logstash-filter-grok>

All the available patterns are in different files which are located here:

<https://github.com/logstash-plugins/logstash-patterns-core/tree/main/patterns/ecs-v1>

Finishing Up

This completes the documentation on How to Configure Filters in Nagios Log Server 2024R2. If you have additional questions or other support-related questions, please visit us at our Nagios Support Forum, Nagios Knowledge Base, or Nagios Library:

[Visit Nagios Support Forum](#)

[Visit Nagios Knowledge Base](#)

[Visit Nagios Library](#)