GUIDE

STORMSHIELD LOG SUPERVISOR

USER MANUAL

Version 1.1.1

Document last updated: August 9, 2023

Reference: sls-en-user_manual-v1.1.1

Table of contents

Change log

Date	Description
August 9, 2022	- Added SES Evolution compatibility
February 10, 2023	- SLS 1.1.1 Release

1. Getting started

Welcome to the Stormshield Log Supervisor version 1.1.1 User Manual.

This manual is a guide designed to help a user to walk through the basics of SLS after the SLS installation has been completed. This manual can be referred to as a guide if a user needs help while carrying out the day to day function of SLS such as executing a search query, creating/managing dashboards, creating incidents, generating a report, etc.

SLS is a log management solution. It collects streaming data coming from Stormshield Network Security firewalls and Stormshield Endpoint Security Evolution, analyzes it, and provides meaningful insights to your data in real-time. SLS allows tight control over widely distributed enterprise networks from a single location and offers capabilities of synthesizing the underlying risks associated with complex distributed attacks on large networks.

SLS provides the following features:

- Collection
- Storage
- Monitoring
- Alert
- Notification
- Reporting

2. Login

2.1 SLS Authentication

SLS Authentication is the basic user authentication mechanism in SLS. In this method of authentication, you enter your credentials to the system. SLS then verifies the username and password combination from the database and authenticates the login based on the result.

You might get the following error messages during *SLS Authentication*.

Scenario	Error Message
Empty Username, Password, or Verify fields	This field is required
Incorrect Username, or Password	Invalid Username or password
Incorrect Captcha Value	Invalid captcha value

2.1.1 Forgot Password?

In case you forget your password, follow the steps to reset it.

1. Click **Forgot Password?**.
2. Provide a valid **Username**.
3. Click **Send**.

A password reset link is sent to the e-mail address associated with the provided username.

IMPORTANT

The link expires in an hour.

2.2 LDAP Authentication

You can use the **LDAP Authentication** to authenticate your users if your organization uses LDAP. You can configure your machine to pull the user credentials and role-based access control rules from the existing **LDAP Directory**.

The **LDAP Authentication** is automatically installed in the SLS as a plugin in the *LP_Default* application. You need to configure the *LDAP Strategy* in SLS to use the *LDAP service*.

2.2.1 Using LDAP Authentication

1. Enter the URL to the SLS system and press **Enter**.
2. Click the **Other Authentication Options** link at the bottom of the panel.
3. Select **LDAP Authentication**.
4. Enter a **Username** and a matching **Password**.
5. Select a **Domain**.
6. Click **Login**.

3. Search

You can use the **Search** tab to search for the logs in the system. The drop-down menu beside the *Search* tab lists the **History**, **Saved Searches**, **Vendor Searches**, and **Labels**. It also contains the quick links to **Search Labels**, **Saved Searches**, **Vendor Searches**, **Search History**, and **Lists**.

The Search tab also has the **Filter** textbox. You can use it to search the logs using *SLS Queries*.

SLS collects logs using different collectors and fetchers. The logs are then indexed and stored securely until the time specified in the system. You can search the logs using an intuitive query language.

The search results let you power your real-time, self-updating dashboard widgets, create custom reports to monitor various compliance requirements, configure different correlation intelligence, and write alert rules to act on the incidents which require a prompt response.

3.1 The Search Bar

The **Search Bar** lets you search the indexed logs. It consists of:

3.1.1 Query Bar

The **Query Bar** is where you enter the query string. A *Query String* is a logical combination of words, phrases, or field values. You can either type a query string or build it to aggregate different values in the search result and display the result in a suitable graphical format.

3.1.2 Repo selector

The **Repo Selector** lets you select the repositories in which to search the logs. Each repository collects the logs and stores them for a pre-defined period. From the *Repo Selector*, you can select multiple repositories by clicking the drop-down menu on the right. However, for better performance, choose only those repos that are required.

The repos in the **Repo Selector** are grouped either by Distributed SLSs or by Repo. From the *Repo Selector*, click **Change** to open a panel where you can choose to change how the repos are grouped.

3.1.3 Time range

You can select the **Time Range** and apply it to your search. The default *Time Range* is **Last 10 minutes**. You can choose a range in the **Last x** time-range format, or select a range from the **Custom Range** time-range picker to set boundaries for your searches. The custom time-ranges **Last 1 hour**, **Last 6 hours**, **Last 7 days** are available in the drop-down menu.

3.1.4 Use Wizard

Use Wizard is a utility within SLS that guides you through the steps of building a simple search query.

To use **Use Wizard**, follow the steps below:

1. Go to `Search`.
2. Click **Use Wizard** to open the *Search Wizard* panel.

3. Enter the **words/phrases** that you want to search.
4. Enter the **words/phrases** that you want to exclude from the results.
5. Click **Continue**.

> **NOTE**
> You can click **Search Now** at any time while building the search query in this way. It searches for the logs using the query built up to that point in the process.

6. Select a **Visualization** option.
   - Select **Chart** or **Timechart** to open the *Use Suggested Aggregation Functions* panel.
     - Select a **Functions** and fields to **Aggregate On**.
     - Click **Continue**.
     - Choose fields from which to **Group** the results.
     - Click **Search Now**.
   - Select Latest to open the *Group the Latest Result By* panel.
     - Choose fields from which to **Group** the latest results by.
     - Click **Search Now**.
   - Select **Selected Fields** to visualize the search results based on the selected fields.
     - Choose the **Fields** from the drop-down menu.
     - Click **Search Now**.

> **NOTE**
> In the Data Privacy Module enabled systems, you won't be able to view the raw logs.

3.2 Query Language

SLS's **Query Language** is extensive, intuitive, and user-friendly. It covers all the search commands, functions, arguments, and clauses. You can search the log messages in various formats depending on the query you use.

SLS also supports chaining of commands and multi-line queries. Use a pipe (|) to chain the commands and press **Shift + Enter** to add a new line in the query. The search keywords are not case-sensitive.

> **NOTE**
> The examples of some search queries provided in this section may not yield any result as the relevant logs may not be available in your system.

3.2.1 Simple Search

You can use the following types of simple queries to familiarize yourself with the SLS Query Language.

Single word

Single word search is the most basic search that you can run in SLS. Enter a **single word** in the *Query Bar* to retrieve the logs containing the word.

```
login
```

This query searches for all the logs containing the word **login** in the message.

Multiple words

Searching with multiple words lets you search the original logs using a combination of words. For searches with multiple words, only the logs containing all the words are displayed.

> **NOTE**
>
> The order of the words is not important.

```
account locked
```

This query searches for all the logs containing both the search terms **account** and **locked** in the message.

Phrases

Phrase Search lets you search the exact phrase in the logs. You must enclose the words inside double-quotes (" ").

> **NOTE**
>
> The order of the words is important.

```
"account locked"
```

This query searches for all the logs containing the exact phrase **account locked**.

Field values

The normalized logs contain information in key-value pairs. You can use these pairs directly in the log search. To see all the logs from the user **Bob**, use the following query:

```
user = Bob
```

This query searches for all the logs from the user **Bob**.

```
device_ip = 192.168.2.1
```

This query searches for all the logs coming from the device with the IP Address **192.168.2.1**.

You can combine multiple field value pairs as:

```
device_ip = 192.168.2.1 sig_id = 10051
```

You can also combine this with a simple query as:

```
login device_ip = 192.168.2.1 sig_id = 10051
```

Logical operators

You can use various keywords to perform logical operations in the SLS search query.

And

Use the logical operator **and** to search for the slogs containing both the specified parameters.

```
login and successful
```

This query searches for all the messages containing the word *login* **and** the word *successful*.

The **and** operator can also be used for key-value search queries as follows:

```
login and device_ip=192.168.2.2
```

Or

Use the logical operator **or** to search for the logs containing either of the specified parameters.

```
login or logout
```

This query searches for all the messages containing either the word *login* **or** the word *logout*.

This operator can also be used with the key-value search query as follows:

```
device_ip = 192.168.2.1 or device_ip = 127.0.0.1
```

Not

You can use the hyphen (-) symbol for the logical negation in your searches.

```
login -Bob
```

This query searches for the log messages containing the word **login** but not the word **Bob**.

```
-device_ip = 192.168.2.243
```

This query returns the logs containing all the **device_ip**s except **192.168.2.243**.

NOTE

- While searching with field-names, you can also use the **!=** symbol to denote negation.

  ```
  device_ip != 192.168.2.243
  ```

- By default, the **or** operator binds stronger than the **and** operator. Therefore, for the query **login or logout and MSWinEventLog**, SLS returns the log messages containing either **login** or **logout**, but containing **MsWinEventLog**.

Parentheses

In SLS, the **or** operator has a higher precedence by default. You can use parentheses to override the default binding behavior when using the logical operators in the search query.

```
"login failed" or (denied and locked)
```

This query returns the log messages containing *login failed* or both *denied* **and** *locked*.

Wildcards

You can use wildcards as replacements for a part of the query string. Use the following characters as wildcards:

1. **?** - Replacement for single character.
2. ***** - Replacement for multiple characters.

If you want all the log messages containing the word **login** or **logon**, use the following:

```
log?n
```

> **NOTE**
>
> This query also searches for the log messages containing other variations such as **logan**, **logbn**, and **logcn**.

```
log*
```

This query returns the logs containing the words starting with **log** such as **SLS**, **logout**, and **login**.

> **NOTE**
>
> You can also use *Wildcards* while forming a search query with field names. To get all the usernames that end in **t**, use the following.
>
> ```
> username = *t
> ```

Step

You can use the **step** function to group fields. To see the log messages with **destination_port** in steps of 100 as follows:

destination_port	count
0 - 100	50
100 - 200	32

```
step(destination_port,100) = 0 | chart count() by destination_
port
```

This query searches for all the log messages containing the field **destination_port**, and groups them in steps of 100. The value at the end of the query specifies the starting value of the **destination_port** for grouping.

> **NOTE**
>
> You can use the **step** to group using multiple field names.

Lower and Upper

You can change type-case of your field values. Use the **lower** function to change the values to lower case. Similarly, use the **upper** function to change the field values to upper case. The

upper and **lower** functions change the type-case of the values to the same case so that you can observe consistent results.

Use the **upper** and **lower** functions with *chart* and *timechart* commands.

```
| chart count() by upper(action)

| timechart count() by lower(action)
```

Time Functions

The **Time Functions** extract specied values from a time-based field. The following time functions are supported in the *Simple Search Query*:

- second
- minute
- hour
- day
- day of week
- month

The arguments taken by these functions are numeric. These functions parse **Unix Timestamps**.

> **NOTE**
>
> Unix time is a system for describing instants in time, defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970, not counting leap seconds. It is used widely in Unix-like and many other operating systems and file formats.
>
> Example: 1384409898 is the Unix time equivalent of Thu, 14 Nov 2013 06:18:18 GMT

In SLS, **col_ts** and **log_ts** carry Unix timestamps. However, you can create your own fields which contain the Unix timestamps using the **rex** or **norm** commands.

second

You can use the **second** function to search for the logs generated or collected in seconds.

The generic syntax for second is:

```
second(field) = value
```

The value for second ranges from 0 to 59.

```
second(log_ts) = 23
```

This query searches for the logs generated during the twenty third second.

minute

You can use the **minute** function to search for the logs generated or collected in minutes. The values for the minute range from 0 to 59.

```
minute(col_ts) = 2
```

This query searches for the logs generated during the second minute.

minute() can also be used in aggregation functions.

hour

You can use the **hour** function to search for the logs generated or collected in hours. The values for the hour range from 1 to 24.

Example:

```
hour(col_ts) = 1
```

This query displays the logs generated during the first hour.

day

You can use the **day** function to search for the logs generated or collected in days.

Example:

```
day(col_ts) = 4
```

This query displays the logs of the 4th day.

day of week

You can use the **day of week** function to search the logs for the specific day of the week. The value for **day_of_week** ranges from 1 (Sunday) to 7 (Saturday).

Example:

```
day_of_week(col_ts) = 7 OR day_of_week(col_ts) = 1
```

This query displays the logs in off days, i.e, Saturday and Sunday.

month

You can use the **month** function to search the logs generated or collected in months. The value of month ranges from 1 (January) to 12 (December).

Example:

```
month(col_ts) = 6
```

This query displays the log activity for June.

NOTE

You can use the relational operators (>, <, = and !=) with the time commands to create a sensible time-range for your search queries.

The following table summarizes the time functions:

Time functions	Working Examples	Value Range
second	second(col_ts) = 20	0 - 59
minute	minute(col_ts) = 18	0 - 59
hour	hour(col_ts) = 6	0 - 23
day	day(col_ts) = 14	1 - 31
day_of_week	day_of_week(col_ts) = 5	1 - 7 (Sun - Sat)
month	month(col_ts) = 11	1 - 12 (Jan - Dec)

List

You can create a static list with a number of values, and use this list in the search query instead of keying in all the values.

For example, if you create a list **EMPLOYEES** with the names of all the employees in a company, you can check whether a single user has logged into the system using the following query.

```
user in EMPLOYEES action=login
```

The search query matches the value of the field **user** with all the values in the EMPLOYEES list.

> **IMPORTANT**
>
> The name of the list must be provided in uppercase.

You can also use an **Inline List** while executing a search query.

The generic syntax for inline list is:

```
field in [value1, value2,....]
```

which is equivalent to **field = value1 OR field = value2**.

Example:

```
source_port in [21, 53, 88,  123]
```

In cases where the values have multiple words in the inline List, use quotation marks as shown below.

```
event in ["Process completed", "Process accomplished"]
```

Table

Tables are external file-formats which contain the information you may choose to associate with a search result. The file formats supported for the tables are CSV, ODBC, LDAP, and Threat Intelligence. The information obtained is prefixed with the table alias in the log messages.

For example:

IPList is a CSV table containing fields such as *Address*, *IP*, *Name*, and *SN*. To view the content of this external CSV table, use the following query:

```
table "IPList"
```

The following content is displayed:

To view all student entries in a table called **studentResult**, which contains *student_name*, *student_roll*, and *percentage* as fields, use:

```
table "studentResult"
```

To search for all the student entries in the table **studentResult** who have passed with distinction:

```
table "studentResult" percentage >= 80
```

To search for all the student entries in the table **studentResult** who have failed:

```
table "studentResult" percentage < 40
```

> **NOTE**
>
> In the Data Privacy Module enabled systems, when you use the *table* query, you can only see the values of the search results in the encrypted form. You cannot request a decryption for these values.

3.2.2 Aggregators

Aggregation functions are used with the **chart** and the **timechart** commands to aggregate the fields. The search results can be formatted using fields, chart or timechart commands.

chart

With the **chart** command, you can get log messages in the chart form. If you want to see all the messages containing **login** and group them by **device_ip** you can use the following query.

```
login device_ip = * | chart count() by device_ip
```

This query searches for all the log messages containing the word **login**, and groups them by **device_ip**. It then displays the number of log messages for each **device_ip**.

You can also count by multiple fields. The log message count is then displayed for each of the field.

```
login | chart count() by destination_address, destination_port
```

In this case, the count of the log messages for every combination of **destination_address** and **destination_port** is grouped and the corresponding count is shown.

You can use other aggregation functions such as **max** and **min** in place of **count**.

```
connection | chart max(datasize) by source_address
```

```
datasize=*| chart max(datasize) as mx, min(datasize) as mn,
sum(datasize) as sm by source_address limit 15
```

You can also display the chart in different forms such as Column, Bar, Line, and Area.

You can also modify aggregation functions as follows:

```
object = connection | chart count(action=permitted) by source_
address
```

In this query, only the log messages containing **action=permitted** are counted. You can write the same query as:

```
action = permitted object = connection | chart count() by source_
address
```

Multiple counts or other aggregators can be used in a single query string.

```
object = connection | chart count(action=permitted), count
(action=blocked)
by source_address
```

This query displays two columns. The first is the count of the connections with the permitted action and the second is the count of blocked actions.

timechart

You can use the **timechart** command to chart the log messages as a time series data. It first displays the logs according to the time they were collected or generated. Then, it returns the log results according to the collection time stamp (col_ts) or log generation time (log_ts) as selected in the system.

The terms log_ts and col_ts differ in the function.

log_ts	col_ts
Denotes time present in log messages.	Denotes the time when the log was actually collected in SLS.

For example you can timechart all the messages with **login** shown below.

```
login | timechart count()
```

This plots the count of all the messages containing the word **login** into a graph with the horizontal axis as time. The total time-span is the time selected for the search query.

```
| timechart on log_ts count()
```

This query plots the count of the logs based on the **log_ts** field.

You can also use the **timechart** command to plot the data on a fixed time-interval. To have a **timechart** with bars for every 20 minutes, use the following query:

```
login | timechart count() every 20 minutes
```

You can use every x minutes, every x hours, or every x days with the **timechart** command.

NOTE

When the limit of **timechart()** is not specified, the number of bars of the **timechart** depends on the nature of the query.

- The number is always equal to 30 if the time-range is less than 30 units. For example, if you provide a time span of 10 minutes SLS displays 30 bars in the span of 20 seconds.
- If the time-range is greater than 30 units, the number of bars is equal to the time-range. This holds true until the upper limit of the number of bars is reached, which is 59.
- There are also some special cases for the number of graphs. The number of bars is equal to the number of seconds specified, and the time span of 1 day displays 24 bars in the span of one hour.

Aggregation functions are used with the **chart** and the **timechart** commands by joining them with the | symbol.

The following aggregators are available in SLS:

- count()
- distinct_count()
- sum()
- min()
- max()
- avg()
- var()

count()

You can use the **count** function to get the total number of logs in the search results. For example,

```
| chart count()
```

This query displays the total number of log messages in the search results.

```
login | chart count() by device_ip
```

This query searches for all the log messages containing the word **login**. It then groups the logs by their **device_ip**s and shows the count of the log messages for each of the **Device IP**.

You can also give filters to the **count()** function as shown below.

```
login | chart count(event_id = 528) by device_ip
```

This query looks for all the log messages containing the word **login**. It then groups them by their **device_ip** s and shows the count of the messages containing the field value **event_id = 528**.

distinct_count()

You can use the **distinct_count()** function to get the number of distinct count of the object. For example,

```
| chart distinct_count(destination_port) by destination_address
```

In this case, though different ports may have multiple counts, **distinct_count()** returns the count of the distinct ports for every destination address.

If the search results for a particular destination address had the following data:

port	count
21	20
25	30
901	15

The result for the **distinct_count()** is 3 for each of the ports 21, 25 and 901. However, the result of the **count()** is 65.

sum()

You can use the **sum()** function to sum the values of the specified fields.

```
| chart sum(datasize) by device_ip
```

This query displays the sum of all the **datasize** fields for each **device_ip**.

You can also give filters to the **sum()** function.

```
| chart sum(datasize, datasize > 500)
```

This query only sums a datasize if it is greater than 500. The expression can be any valid query string but must not contain any view modifiers.

max() and min()

These functions can be used to find the maximum or minimum value of the specified field.

```
| chart max(severity) by device_ip
```

This query displays the maximum **severity** value in each of the **device_ip**.

```
login | chart count(), max(col_ts) by device_ip, col_type
```

This query looks for all the log messages containing the word **login**. Then, it groups the search results by their **device_ip**s and the **col_type** and shows the count of the log messages and the latest **col_ts** for each of the groups.

The **max()** and **min()** functions also support filter expressions as:

```
| chart max(severity, severity < 5)
```

This query shows the maximum **severity** that is less than 5.

avg()

You can use the **avg()** function to calculate the average of all the values of the specified field.

```
| chart count(), avg(response_time, response_time=*)
```

This query calculates the average **response_time**.

var()

You can use the **var()** function to calculate the variance of the field values. Variance describes how far the values are spread out from the mean value.

Execute the following query for proper visualization of how the data fluctuates around the average value.

```
severity = * | chart count(),avg(severity),var(severity) by
device_ip
```

> **NOTE**
>
> You can use **+**, **-**, *****, **/**, and **^** to add, subtract, multiply, divide, and to raise the power in the **min()**, **max()**, **sum()**, **avg()**, and **var()** functions.

Example:

```
avg(field1/field2^2+field3)
```

IMPORTANT

While using the expressions such as **avg()**, and **min()**, it is good to use a proper filter to discard log messages not containing the specified fields.

distinct_list()

You can use the **distinct_list()** function to return the list of all the distinct values of the field. For example, if you want to view all the distinct values of the field *action* in the system, you can use the following query:

```
| chart distinct_list(action)
```

You can use a grouping parameter to group the distinct list. For example:

```
| chart distinct_list(action) as actions by user
```

The above query returns the list of every distinct value of the field *action* in the *actions* column grouped by the grouping parameter *user*. You can use this example to view all the actions performed and machines used by every user in the system.

You can also use this function with other aggregation functions. For example:

```
user=Jolly | chart distinct_list(action) as actions, distinct_
count(action) as actions_count by user
```

The above query returns the list of all the distinct actions with their counts for the user *Jolly*.

3.2.3 One-to-One Commands

The **One-to-one** commands take one value as input and provide one output.

For example, you can use the **rex** and the **norm** commands to extract specific parts of the log messages into an ad-hoc field name. This is equivalent to normalizing log messages during the search. However, the extracted values are not saved.

The **rex** and **norm** commands do not filter the log messages. They list all the log messages returned by the query and add the specified ad-hoc key-value pairs if possible.

rex

You can use the **rex** command to recognize regex patterns in the **re2** format. The extracted variable is retained only for the current search scope. The result also shows the log messages that are not matched by the rex expression.

Example Log:

```
Oct 15 20:33:02 WIN-J2OVISWBB31.immuneaps.nfsserver.com
MSWinEventLog   1   Security    169978  Sat Oct 15 20:33:01 2011
5156    Microsoft-Windows-Security-Auditing N/A N/A Success Audit
WIN-J2OVISWBB31.immuneaps.nfsserver.com Filtering Platform
Connection   The Windows Filtering Platform has allowed a
connection.    Application Information:    Process ID:  4
Application Name: System   Network Information:    Direction:
Inbound Source Address:  192.168.2.255   Source Port:  138
Destination Address: 192.168.2.221   Destination Port:  138
Protocol:  17    Filter Information: Filter Run-Time ID: 67524
```

```
Layer Name:  Receive/Accept   Layer Run-Time ID: 44 169765
```

You can use the **rex** command to extract the protocol id into a field protocol_id with the following syntax:-

```
| rex Protocol:\s*(?P<protocol_id>\d+)
```

The query format is similar to the following:

```
| rex any regular expression:\s+(?P<field_name>expression to
capture to field)
```

IMPORTANT

The (?P< >) expression is part of the rex syntax to specify the field name.

You can also extract multiple fields from a single rex operation as shown below.

```
| rex Source Address:\s*(?P<src_address>\d+\.\d+\.\d+\.\d+)
```

The extracted values can be used to chart your results. For example,

```
| rex Protocol:\s+(?P<protocol_id>\d+) | chart count() by
protocol_id
```

Since the rex command acts on the search results, you can add it to a query string as shown below:

```
Windows Filtering AND allowed | rex Protocol:\s+(?P<protocol_
id>\d+)
```

```
user=* | rex on user:\s+(?P<account>\S+)@(?P<domain>\S+) |
chart count() by account, domain | search account=*
```

NOTE

Use Single quote to address inline normalization while using square bracket. For example:

This syntax works: `| norm on user <my_user:\S+> | chart count() by my_user`.

But this does not. `| norm on user <my_user:[A-Z]+> | chart count() by my_user`.

If you use the box brackets (**[**, **]**), single quote (') is necessary in the syntax.

norm

You can use the **norm** command to extract variables from the search results into a field. The difference between the **rex** command and the **norm** command is that **norm** supports both, SLS normalization syntax and re2 syntax, whereas the **rex** command only supports re2 syntax.

Example Log:

```
Dec 17 05:00:14 ubuntu sshd[7596]: Invalid user Bob from
110.44.116.194
```

To extract the value of the user into the field *user*, use the following syntax:-

```
| norm Invalid user <user:word>
```

And this can also be used to chart in the graph as follows.

```
| norm Invalid user <user:word>| chart count() by user
```

You can also use the **norm** command to extract multiple key-value pairs as shown below:

```
| norm Invalid user <user:word> from <source_ip:ip>
| chart count() by my_user, msg | search my_user=*
```

NOTE

- For the list of definers (simplified regular expressions), refer to the appendix of the **User Manual**.
- Use Single quote to address inline normalization while using square bracket. For example:
  This syntax works: `| norm on user <my_user:\S+> | chart count() by my_user`.
  But this does not. `| norm on user <my_user:[A-Z]+> | chart count() by my_user`.
  If you use the box brackets (**[**, **]**), single quote (') is necessary in the syntax.

fields

You can use the **fields** command to display the search results in a tabular form. The table is constructed with headers according to the field-names you specify. SLS returns **null** if the logs do not contain the specified fields.

```
| fields source_address, source_port, destination_address,
destination_port
```

Here, the fields **source_address**, **source_port**, **destination_address**, and **destination_port** are displayed in a tabular form as shown above.

Any log message without the field **destination_port** has a corresponding row with the **destination_port** column value as -N/A-.

rename

You can use the **rename** command to rename the original field names.

Example:

```
| rename device_ip as host
```

When multiple fields of a log are renamed as the same name, the rightmost field takes precedence over others and only that field is renamed.

Example:

```
| rename source_address as ip, destination_address as ip
```

Here, if both the *source_address* and *destination_address* fields are present in a log, only the *destination_address* field is renamed as *ip* in search results.

The log messages after normalization can have different field-names for information carrying similar values. For example, different logs may have **name**, **username**, **u_name**, or **user_name** as keys for the same field username. To aggregate all the results and analyze them properly, you can use the rename command.

```
| rename target_user as user, caller_user as user | chart count()
by user
```

In some cases, the field names can be more informative with the use of rename command as below:

```
label = Attack | rename source_address as attacking_ip |
chart count() by attacking_ip
```

process

You can use the **process** command to execute different one-to-one functions which produce one output for one input given.

Some default process commands available in SLS are:

String Concat

This process command lets you join multiple field values of the search results.

Syntax:

```
| process concat(fieldname1, fieldname2, ...., fieldnameN) as
string
```

Example:

```
| process concat(city, country) as geo_address
```

Domain Lookup

This process command provides the domain name from a URL.

Syntax:

```
| process domain(url) as domain_name
```

Example:

```
url=* | process domain(url) as domain_name |
chart count() by domain_name, url
```

Difference

This process command calculates the difference between two numerical field values of a search.

Syntax:

```
| process diff(fieldname1,fieldname2) as string
```

Example:

```
| process diff(sent_datasize,received_datasize) as difference
| chart count() by sent_datasize, received_datasize,difference
```

Summation

This process command calculates the sum between two numerical field values of a search.

Syntax:

```
| chart sum(fieldname)
```

Example:

```
label = Memory | chart sum(used) as Memory_Used by col_ts
```

Experimental Median Quartile Quantile

The Experimental Median Quartile Quantile process command performs statistical analysis (median, quartile, and quantile) of events based on fields. All these commands take numerical field values as input.

Median

Syntax:

```
| chart median(fieldname) as string
```

Example:

```
doable_mps=* |chart median(doable_mps)
```

Quartile

Syntax:

```
| chart quartile(fieldname) as string1, string2, string3
```

Example:

```
doable_mps=* |chart quartile(doable_mps)
```

Quantile

Syntax:

```
| process quantile(fieldname)
```

Example:

```
doable_mps=* | process quantile(doable_mps)
|search quantile>0.99
|chart count() by doable_mps order by doable_mps desc
```

Process lookup

This process command looks up for the related data from the user defined table.

Syntax:

```
| process lookup(table,field)
```

Example:

```
| process lookup(lookup_table, device_ip)
```

GEOIP

This process command gives the geographical information of a public IP address. It adds a new value "internal" to all the fields generated for the private IP supporting the RFC 1918 Address Allocation for Private Internets.

Syntax:

```
| process geoip (fieldname)
```

Example:

```
| process geoip (source_address)
```

For the Private IP:

For the Public IP:

Codec

The Codec process command encodes the field values to ASCII characters or decodes the ASCII characters to their text value.

Syntax:

```
| process codec(<encode/decode function>, <field to be
encoded/decoded>) as <attribute_name>
```

Example:

```
| process codec(encode, name) as encoded_name
```

InRange

The InRange process command determines whether a certain field-value falls within the range of two given values. The processed query returns TRUE if the value is in the range.

Syntax:

```
| process in_range(endpoint1, endpoint2, field, result,
inclusion)

where,

endpoint1 and endpoint2 are the endpoint fields for the range,
the field is the fieldname to check whether its value falls
within the given range,
result is the user provided field to assign the result (TRUE or
FALSE),
inclusion is the parameter to specify whether the range is
inclusive or exclusive of
given endpoint values. When this parameter is TRUE, the endpoints
will be included for
the query and if it is FALSE, the endpoints will be excluded.
```

Example:

```
| process in_range(datasize, sig_id, duration,Result, True)
```

Regex

The Regex process command extracts specific parts of the log messages into custom field names.

Syntax:

```
| process regex("_regexpattern", _fieldname)
| process regex("_regexpattern", "_fieldname")

Both syntaxes are valid.
```

Example:

```
| process regex("(?P<type>\S*)",msg)
```

DNS Process

This process command returns the domain name assigned to an IP address and vice-versa. It takes an **IP address** or a **Domain Name** and a **Field Name** as input. The plugin then verifies the value of the field. If the input is an *IP Address*, it resolves the address to a *hostname* and if the input is a *Domain Name*, it resolves the address to an *IP Address*. The output value is stored in the *Field Name* provided.

Syntax:

```
| process dns(IP Address or Hostname)
```

Example:

```
destination_address=* | process dns(destination_address) as
domain
| chart count() by domain
```

Compare

This process command compares two values to check if they match or not.

Syntax:

```
| process compare(fieldname1,fieldname2) as string
```

Example:

```
| process compare(source_address, destination_address) as match
| chart count() by match, source_address, destination address
```

IP Lookup

This process command enriches the log messages with the Classless Inter-Domain Routing (CIDR) address details. A list of CIDRs is uploaded in the CSV format during the configuration of the plugin. For any *IP Address* type within the log messages, it matches the IP with the content of the user-defined Lookup table and then enriches the search results by adding the CIDR details.

Syntax:

```
| process ip_lookup(IP_lookup_table, column, fieldname)
 where IP_lookup_table is the lookup table configured in the
plugin,
 Column is the column name of the table which is to be matched
  with the fieldname of the log message.
```

Example:

```
| process ip_lookup(lookup_table_A, IP, device_ip)
```

This command compares the IP column of the lookup_table_A with the device_ip field of the log and if matched, the search result is enriched.

Compare Network

This process command takes a list of *IP addresses* as inputs and checks if they are from the same network or different ones. It also checks whether the networks are public or private. The comparison is carried out using either the default or the customized CIDR values.

Syntax:

```
| process compare_network(fieldname1,fieldname2)
```

Example: (Using default CIDR value)

```
source_address=* destination_address=*
| process compare_network (source_address, destination_address)
| chart count() by  source_address_public, destination_address_
public,
same_network, source_address, destination_address
```

Clean Char

This process command removes all the alphanumeric characters present in a field-value.

Syntax:

```
| process clean_char(<field_name>) as <string_1>, <string_2>
```

Example:

```
| process clean_char(msg) as special, characters
| chart count() by special, characters
```

Current Time

This process command gets the current time from the user and adds it as a new field to all the logs. This information can be used to compare, compute, and operate the timestamp fields in the log message.

Syntax:

```
| process current_time(a) as string
```

Example:

```
source_address=* | process current_time(a) as time_ts
| chart count() by time_ts, log_ts, source_address
```

Count Char

This process command counts the number of characters present in a field-value.

Syntax:

```
| process count_char(fieldname) as int
```

Example:

```
| process count_char(msg) as total_chars
| search total_chars >= 100
```

DNS Cleanup

This process command converts a DNS from an unreadable format to a readable format.

Syntax:

```
| process dns_cleanup(fieldname) as string
```

Example:

```
col_type=syslog | norm dns=<DNS.string>| search DNS=*
|process dns_cleanup(DNS) as cleaned_dns
| norm on cleaned_dns .<dns:.*>.
| chart count() by DNS, cleaned_dns, dns
```

Grok

The **grok** process command enables you to extract key-value pairs from logs during query runtime using Grok patterns. Grok patterns are the patterns defined using regular expression that match with words, numbers, IP addresses, and other data formats.

You can find a list of all the Grok patterns and their corresponding regular expressions from the Appendix: Grok Patterns section.

Syntax:

```
| process grok("<signature>")
```

A **signature** can contain one or more Grok patterns.

Example:

To extract the IP address, method, and URL from the log message:

```
192.168.3.10 GET /index.html
```

Use the command:

```
| process grok("%{IP:ip_address_in_log} %{WORD:method_in_log} %
{URIPATHPARAM:url_in_log}")
```

Using this command adds the **ip_address_in_log**, **method_in_log**, and **url_in_log** fields and their respective values to the log if it matches the signature pattern.

AsciiConverter

This process command converts hexadecimal (hex) value and decimal (dec) value of various keys to their corresponding readable ASCII values. The application supports the Extended ASCII

Table for processing decimal values.

Hexadecimal to ASCII

Syntax:

```
| process  ascii_converter(fieldname,hex) as string
```

Example:

```
| process ascii_converter(sig_id,hex) as alias_name
```

Decimal to ASCII

Syntax:

```
| process  ascii_converter(fieldname,dec) as string
```

Example:

```
| process ascii_converter(sig_id,dec) as alias_name
```

WhoIsLookup

The whoislookup process command enriches the search result with the information related to the given field name from the WHOIS database.The WHOIS database consists of information about the registered users of an Internet resource such as registrar, IP address, registry expiry date, updated date, name server information and other information. If the specified field name and its corresponding value are matched with the equivalent field values of the WHOIS database, the process command enriches the search result, however, note that the extracted values are not saved.

Syntax:

```
| process whoislookup(field_name)
```

Example:

```
domain =* | process whoislookup(domain)
```

Eval

This process command evaluates mathematical, boolean and string expressions. It places the result of the evaluation in an identifier as a new field.

Syntax:

```
| process eval("identifier=expression")
```

Example:

```
| process eval("Revenue=unit_sold*Selling_price")
```

toList

This process command populates the dynamic list with the field values of the search result.

Syntax:

```
| process toList (list_name, field_name)
```

Example:

```
device_ip=* | process toList(device_ip_list, device_ip)
```

toTable

This process command populates the dynamic table with the fields and field values of the search result.

Syntax:

```
| process toTable (table_name, field_name1, field_name2,....,
field_name9)
```

Example:

```
device_ip=* | process toTable(device_ip_table, device_name,
device_ip, action)
```

3.2.4 Filtering Commands

Filtering commands help you filter the search results.

search

Using the **search** command, you can conduct searches on the search results. The SLS search query searches on dynamic fields returned from the **norm**, **rex**, and the **table** commands.

To search for users who have logged in more than 5 times:

```
login user = * | chart count() as count_user by user | search
count_user > 5
```

If you create a dynamic field **new field** using norm command as,

```
| norm actual_mps = < new_field:int >
```

To view the logs which have **100** as the value of the new field, use the search command as:

```
| norm actual_mps = < new_field:int >|search new_field = 100
```

We recommend you to use the **search** command only in the following cases:

- When you need to filter the results for simple search (non key-value search).
  For example:

  ```
  | search error
  ```

- When you need to filter the results using the **or** logical operator.
  For example:

  ```
  | search device_name=localhost or col_type=filesystem
  ```

> **NOTE**
>
> It is not advised to use the search command unless absolutely necessary. The reason for this is that the search command uses heavy resources. So, it is always better to apply any kind of filtering before using the search command.

filter

The **filter** command lets you further filter the logs retrieved in the search results.

Syntax:

```
<search query> | filter <condition>
```

For example, if you want to display only the domains that have more than 10 events associated with them in the search results, use the following query:

```
norm_id=*Firewall url=* | process domain(url) as domain | chart
count() as events by domain | filter
events>10
```

The query searches for all the logs containing the fields **url** and **norm_id** with the value of **norm_id** having *Firewall* at the end. It then adds a new field **domain** to the logs based on the respective URLs and groups the results by their domains. Finally, the **filter** command limits the results to only those domains that have more than 10 events associated with them.

The **filter** command does not index the intermediate fields, and thus, is computationally more efficient than the **search** command. Therefore, SLS uses the **filter** command to drill-down on the search results, which significantly speeds up the drill-down process.

> **NOTE**
>
> - The **filter** command filters the results based on dynamic fields returned from the **norm**, **rex**, and **table** commands as well.
> - The **filter** command only works with expressions having the =, >, <, >=, and <= operators.
> - To filter the results with more than one condition, you must chain multiple **filter** expressions.

latest

The **latest** command finds the most recent log messages for every unique combination of provided field values.

```
| latest by device_ip | timechart count() by device_ip
```

This query searches for the latest logs of all the devices.

```
status = down port = 80 | latest on log_ts by device_ip
```

This query searches for all the latest devices based on the **log_ts** field whose web server running on the port number 80 is down.

order by

Use the **order by** to sort the search results based on a numeric field. You can sort the results in either the *ascending* or the *descending* order.

Examples:

```
device_name= "John Doe" and col_type="syslog" | order by col_ts
asc
```

This query searches for all the syslog messages generated from the device named **John Doe** and sorts them in the ascending order of their **col_ts** values.

```
device_name=* | order by log_ts desc
```

This query searches for the logs from all the devices in the system and sorts them in the descending order of their **log_ts** values.

limit <number>

Use the **limit <number>** command to limit the number of results displayed. Additionally, you can add the **other** keyword at the end of the query to display the aggregation of the rest of the results.

> **NOTE**
>
> - The feature to display the *Top-10 and the Rest* graphs is supported for the aggregation queries.
> - While using the limit command to retrieve a large volume of logs, make sure that your system has enough resources to load and render the data.

Example:

```
destination_address = * | chart count() by source_address limit
10 other
```

This query searches for all the logs having a **destination address**, filters the top 10 results by their **source address** and rolls-up all the remaining results in the eleventh line. The **source_address** field displays the word *other* in the table as shown in the figure below.

Some other working examples:

```
device_ip=*| chart count() by action, source_address limit 5
other

| chart sum(actual_mps) by service limit 20 other

| chart count() by action limit 10 other
```

3.2.5 Pattern Finding

Pattern finding is a method of finding one or multiple streams and patterns of data to correlate a particular event. For example: five failed logins, followed by a successful login. It can be performed on the basis of the count and the time of occurrence of the stream. Use the *Pattern Finding* rules to detect complex event patterns in a large number of logs.

Correlation is the ability to track multiple types of logs and deduce meanings from them. It lets you look for a collection of events that make up a suspicious behavior and investigate further.

Single Stream

A stream consists of a count or occurrence of a query. The query can be a simple search query or an aggregating query. The stream can consist of a **having same** or a **within** expression. Stream has notion of time.

Syntax	Description
[ ]	For single streams, square brackets contain a stream of events.
within	Keyword to denote the notion of time frame
having same	Keyword

Following are the working examples for pattern finding using single stream:

To find 5 login attempts:

```
[5 action = "logged on"]

[5 login]
```

To find 5 login attempts within a timeframe of 2 minutes:

```
[5 action = "logged on" within 2 minutes]

[5 login within 2 minutes]
```

To find 5 login attempts by the same user:

```
[5 action = "logged on" having same user]

[5 login having same user]
```

To find 10 login attempts by the same user from the same source_address (multiple fields) within 5 minutes:

```
[10 action = "logged on" having same user, source_address within
5 minutes]
```

The time format for specifying timeframe are: second(s), minute(s), hour(s) and day(s).

```
[error] as E
```

This query finds the logs with errors. It then aliases the result as E and displays the fields prefixed with E such as E.severity, and E.device_ip. You can then use the aliased fields as shown below:

```
[error] as E | rename E.device_ip as DIP | search DIP =
"127.0.0.1"
```

Pattern finding queries for different conditions:

10 login to localhost (source_address) by the same user for the last 15 minutes.

```
[10 login source_address = 127.0.0.1 having same user_name within
15 minutes]
```

The field of a log file with a norm command .

```
[2 login | norm <username:word> login successful having same
username within 10 seconds]
```

Multiple Streams

You can join multiple patterns by using **Pattern Finding by Joining Streams** and **Pattern Finding by Following Streams**.

Left Join

You can use a left join to return all the values from the table or stream on the left, and only the common values from the table or stream on the right.

Example:

```
[table event_prob] as s1
left join [event = * | chart count() by event] as s2
on s1.event = s2.event
```

Right Join

You can use a right join to return all the values from the table or stream on the right and only the common values from the table or stream on the left.

Example:

```
[5 transaction error having same user within 30 seconds] as s1
right join [transaction successful] as s2
on s1.user=s2.user
```

Join

Join queries are used to link the results from different sources. The link between two streams must have an **on** condition. The link between two lookup sources or any of the lookup and stream does not require a time-range. Join as a part of a search string, can link one data-set to another based on one or more common fields. For instance, two completely different data-sets can be linked together based on a username or event ID field present in both the data-sets.

The syntax for joining multiple patterns is as follows:

[stream 1] <aliased as s1> <JOIN> [stream 2] <aliased as s2> on <Join_conditions> | additional filter query.

```
[action = locked] as locked
join
[action = unlocked] as unlocked
on
locked.target_user = unlocked.target_user
| chart count() by locked.target_user, locked.caller_computer,
unlocked.caller_user
```

```
[login] as l join [table User] as u on l.user = u.user
```

To find the events where a reserved port of an Operating System (inside the PORT_MACHINE table) is equal to the blocked port (inside the BLOCKED_PORT table):

```
[table PORT_MACHINE port<1024] as s1 join [table BLOCKED_PORT] as
s2
on s1.port=s2.port
```

To find 5 login attempts by the same user within 1 minute followed by 5 failed login attempts by the same user within 1 minute

```
[5 login having same user within 1 minute] as s1
followed by
[5 failed having same user within 1 minute]
```

To find 5 login attempts by the same user within 1 minute followed by 5 failed attempts by the same user within 1 minute and users from both result are same

```
[5 login having same user within 1 minute] as s1
followed by
[5 failed having same username within 1 minute] as s2
on
s1.username = s2.username
```

Followed by

Pattern Finding by *followed by* is useful when two sequential streams are connected to an action.

For example:

```
[2 login success having same user] AS stream1
followed by
[login failure] as stream2
ON
stream1.user = stream2.user
```

Here,

Syntax	Description
[ ] AS stream1	A simple pattern finding query aliased as stream1
followed by	Keyword
[ ] AS stream2	A simple search aliased as stream2
ON	Keyword
stream1.user = stream2.user	Matching field from the 2 streams

The syntax for joining multiple patterns is as follows:

- [stream 1] <aliased as s1> <followed by> [stream 2] <aliased as s2> <within time limit> on <Join_conditions>| additional filter query.
- [stream 1] as s1 followed by [stream2] as s2 within time_interval on s1.field = s2.field
- [stream 1] as s1 followed by [stream2] as s2 on s1.field = s2.field
- [stream 1] as s1 followed by [stream2] as s2 within time_interval

The inference derived from the above queries:

- Streams can be labeled using alias. Here, the first stream is labeled as s1. This labeling is useful while setting the join conditions in the join query.
- The operation between multiple streams is carried out using “followed by” or “join”.
- Use the **followed by** keyword to connect two sequential streams anticipating an action, e.g., multiple login attempts followed by successful login.
- Use the **join** keyword to view additional information in the final search. The **join** syntax is mostly used with tables for enriching the data.
- Time limit for occurrence can also be specified.
- If you use the **join** keyword, then specify the **on** condition.
- Join conditions are simple mathematical operations between the data-sets of two streams.
- Use additional filter query to mitigate false positives which are generally created while joining a stream and a table. Searching the query with a distinct key from the table displays an error-less result.

```
[| chart count() by device_ip] AS lookup
JOIN
[device_ip=*] AS log ON lookup.device_ip = log.device_ip
```

This query does not display histogram but displays the log table.

```
[device_ip=*] as log join  [| chart count() by device_ip] as
lookup on
log.device_ip=lookup.device_ip
```

This query displays both the histogram and the log table.

NOTE

- The **Latest** command is supported in pattern finding queries.
- All the reserved keywords such as **on**, **join**, **as**, and **chart** are not case-sensitive.
- If you want to use reserved keywords in simple search or some other contexts, put them in quotes.

```
login | chart count() by device_ip | search "count()" >
5
```

3.3 Features of the Query Bar

3.3.1 Chaining of commands

You can chain multiple commands into a single query by using the pipe (|) character. Any command except **fields** can appear before or after any other command. The **fields** command must always appear at the end of the command chain.

Example:

```
| chart count() as cnt by device_name | search cnt > 1000
```

This query displays the number of logs with the same **device_name** appearing more than 1000 times.

```
(label = logoff) AND hour (log_ts) > 8 AND hour (log_ts) <16 |
latest by user |
timechart count() by user
```

This query captures all the log messages labeled as **logoff** and those collected between 8 AM and 4 PM. It then displays the timechart of the recent users for the selected time-frame.

3.3.2 Multi-line queries

You can write queries in multiple lines in the SLS Query Bar. Press **Shift + Enter** to add a new line.

Using multiple lines, you can easily visualize long queries. Additionally, it helps you keep track of all the sub-queries by emphasizing corresponding pairs of brackets.

NOTE

- The query bar continues to expand vertically for up to 15 lines. After that, a scroll bar appears to the right.
- Move the text-cursor to a bracket to find its pair.

3.4 Tools

The search result also has a tool bar that gives you an easy access to the various functions right after search.

3.4.1 Found

This is the total number of results found for the search query. SLS searches results in an incremental basis, so this number keeps getting updated until all the results have been fetched.

3.4.2 Search Help Text

In the Search Query bar, if you click the down arrow key, a pop-up panel appears. It contains texts to help you write valid search queries. An alternative way to access this feature is to click *CMD + right click* (on a Mac) and *CTRL+ right click* (on a Windows Machine). You can conduct a search query while simultaneously looking at the search help-text window.

NOTE

The search help text is not displayed if you have disabled the **Display search help pop-up** in `My Preferences >> Search`. For details, refer to My Preferences.

3.4.3 Add Search To

The **Add Search To** option lets you work on or view the results of any search query by forwarding them to various places.

These places are described below:

Add Search To Dashboard

The **Add Search To Dashboard** option lets you create a dashboard widget from a recent search query.

1. Click **Dashboard** to open the *Create Widget* panel.
2. Enter the details for the widget and click **Next**.
3. Select a **Dashboard**.
4. Click **Finish**.

> **NOTE**
>
> The display widgets such as bar graphs, donut charts, and tables automatically appear according to the nature of the result of the search query you enter.

Add Search To Alert Rule

1. Click **Alert Rule** to open the *Create Alert* panel.
2. Provide the **Name**, the **Description**, the **Repos**, and the **Time Range** and click **Next**.
3. Select the **Condition**, the **Risk**, and the **Risk Calculation Function** and click **Next**.
4. Choose a medium for the alert notification.
5. Click **Finish**.

Add Search To Labelling Rule

1. Click **Labelling Rule** to open the *Search Label* panel.
2. Select a **Package** and enter a **List of Labels**.
3. Click **Submit**.

Add Search To Incident

1. Click **Incident** to open the *Create Search Incident* panel.
2. Provide the **Incident Name**, the **Description**, and the **Risk** level.
3. Provide the necessary *Ownership* information.
4. Click **Submit**.

Refer to the **Incidents** section in the **User Manual** for details on creating, managing, and filtering Incidents.

Add Search To Public URL

The **Add Search to Public URL** option lets you add and share Dashboard widgets publicly.

1. Click **Public URL** to open the *Register to Public URL* panel.
2. Specify a **Name**, an **Identifier**, and a **Package** to add your search to a public URL.
3. Click **Ok**.

3.4.4 More

The **More** option lists all the functions that can be carried out for the result of a query.

Export Logs

Export Logs lets you export the search results to the specified target on a remote machine. To export the logs of simple search queries, follow these steps:

1. Go to `Search`.
2. Enter a **Search Query** in the query bar and click **Search**.
3. Click **Export Logs** to open the *Export Options* panel.
4. Specify the **Job Name**, the **Timeout** in seconds, the **Target** and the **Max File Size**.
5. Click **Submit**.

> **NOTE**
>
> The **Export Logs** feature can only be used for simple queries. For aggregated queries, use the **Export as CSV** and **Export as Excel** options.

Permalink

Permalink gives you a complete URL required to generate the current search. You can share this link to other users in the system to make exact and similar search.

Save Search

Save Search lets you save your current search. You can view the *Saved Searches* on the `Search` page under the **My Saved Searches** section.

Report

You can click this option to generate the report of the current search result. For details, refer to Creating a Report from a Search Query.

3.4.5 Stop/Pause

You can pause or stop the search using the corresponding buttons.

3.5 Extended Visualization Framework

In the earlier versions, SLS used the chart modules of ExtJS for visualization of the search results. However, from v5.5.0, SLS also uses D3.js, and plottable.js libraries to create new and visually interactive charts for the search results. SLS is now equipped with even more intuitive features that help you by providing more profound insights into optimizing the available resources, predict failures, foresee trends and tendencies, discover different patterns, and prevent potentially critical incidents altogether.

SLS provides you with an array of visualization options. In addition to the regular bar, line, and column charts, various other statistical tools have been added. Addition of the charts has not only added an aesthetic appeal to the search results but has also helped ease the data analytics process.

You can observe the new visualizations in:

1. Search Interface
2. Dashboards (Widgets)
3. Search Templates

3.5.1 Features of New Visualization

Some features of the new visualization of search graphs are provided below:

1. The legend for the search results is interactive in multiple ways.
   - You can toggle the display of the legend as **ON** and **OFF**. Click the desired legend to hide/unhide it. For example,
     Click the legend of *count()* to hide its corresponding result.
     Click the legend of *avg(datasize)* to hide its corresponding result.
     The status of the legend (either ON or OFF) is saved for the result which is dynamically populated in the widgets in the dashboards.
   - All the related data can be highlighted at once by hovering over the legend. If you hover the mouse over the legend of *count()* then all the data of average is highlighted.
2. The *Pan* and *Zoom* feature in the axes is provided for better visibility of the results.
   - **Pan** is the ability to click and drag the cursor over the search result visualization to select the desired area. With this feature, the axes can be moved to cover a larger area of the timespan of the search result.
   - **Zoom** is the ability to expand and shrink the scale of the axis. With this feature, the scale of the axes can be zoomed in and out for better visibility of the search results.
3. In the **Timechart** responses, a new feature called "Drilldown via Drag Box" has been introduced. If you click and drag the mouse inside the container, a transparent drag box appears. This drag box is movable and resizable.
   The main purpose of the drag box is to further drill-down within a custom time-range which is a subset of the previous time-range. Once the desired vicinity of the drag box is set, click the drill-down icon. This displays the search results of the particular time-frame tapped by the drag box.
4. The axes label auto-adjusts as per the size of the container. This feature is especially useful for dashboards with many widgets where the size of a widget is user-configurable.
   Whenever you resize a widget or click the *Auto-arrange* option, the labels of both the axes auto-adjust as per the space occupied by the search graph whenever applicable.
   Consider the third widget (Multiple Aggregation with Grouping) of the following dashboard:
5. The legend's text auto adjusts as per the widget's dimension. When the container's dimension is expanded or shrunk, the legend's text auto-adjusts without blocking the search result.
   As you customize the size of a widget in the first row, you can see that the legend of the donut chart automatically adjusts.

3.5.2 Response Types in Visualization

Altogether, there are eight response types for the representation of search results in the visualization. Four of them are the regular response types, and the other four response types are the same responses grouped into time buckets for a given time-range.

1. Single Aggregation without Grouping
2. Single Aggregation with Grouping
3. Multiple Aggregation without Grouping

4. Multiple Aggregation with Grouping
5. Timechart Single Aggregation without Grouping
6. Timechart Single Aggregation with Grouping
7. Timechart Multiple Aggregation without Grouping
8. Timechart Multiple Aggregation with Grouping

Single Aggregation without Grouping

The Single Aggregation without Grouping response type is used for aggregation of an individual parameter concerning a given aggregation parameter.

The general syntax for the **Single Aggregation without Grouping** is:

```
| chart aggregation_parameter
```

This search query displays the value of the aggregation parameter over a specified range of time. The result of this response type can be represented in the form of **Display** and **Gauge** charts.

Display

The **Display** format shows the value of the aggregation parameter in the container.

To view the search results in display format, select *Display* from the drop-down menu on the top-right corner of the **Search Result** page.

> **NOTE**
>
> By default, SLS renders the display format for all queries of the *Single Aggregation without Grouping* type.

Gauge chart

Gauge chart, also known as speedometer chart, uses a single needle to show the information as a reading on a dial. The graph is used to visualize percentage values as well as a fixed range of data.

To view the search results in a Gauge chart, select *Gauge* from the drop-down menu on the top-right corner of **Search Result** page.

Click the settings icon on the right side of the chart container to open the **Rendering Parameters** panel.

The value of the aggregation parameter determines the value pointed by the needle. You can configure the maximum value of the dial from *Max value* while rendering parameters. When a value of the aggregation parameter is equal to, or greater than the *Max value*, the percentage value of the needle is displayed as *100%*.

Three different colors, green, yellow, and red are used to represent the limits for the data being depicted in the gauge. By default, the green, yellow, and red colors represent the low, mid, and high range of values respectively. However, you can configure the threshold value (in percentage) to display the dial in the yellow and red colors.

You can specify the threshold value for the red and yellow colors in the **Red Starts** and **Yellow Starts** configuration fields on the **Rendering Parameters** panel.

> **NOTE**
>
> The value of **Red starts** is 90% and **Yellow starts** is 70% by default.

Single Aggregation with Grouping

The **Single Aggregation with Grouping** response type is used for aggregation of various grouping parameters concerning a given aggregation parameter. The general syntax for *Single Aggregation with Grouping* is:

```
| chart aggregation_parameter by grouping_parameter1, grouping_
parameter2, ........, grouping_parametern
```

Example queries of Single Aggregation with Grouping are:

```
| chart count() by device_name
```

```
| chart sum(datasize) by action, protocol
```

```
| chart avg(datasize) by type, protocol, device_ip
```

The response type displays the value of the aggregation parameter, grouped by all the grouping parameter(s) in the specified time range. The result of this query can be represented in the form of **Column**, **Line**, **Donut**, **Area**, **Bar**, **Heatmap**, **Radar**, **Treemap**, **Parallel Coordinate**, **Sankey**, **World Map**, and **ATT&CK** charts.

General Operations for Single Aggregation with Grouping

This section contains the general operations applicable to all the charts belonging to the *Single Aggregation with Grouping* response type.

> **NOTE**
>
> Some charts might consist of operations that are relevant to the specific chart only. In that case, refer to the section of the particular chart.

Drill-down

In the Single Aggregation with Grouping response type, you can perform the drill-down specific value of the grouping or aggregation parameter.

When you hover over a component of a graph (including but not limited node, line, bar, point) a tooltip appears. The tooltip displays all the relevant information about the particular component.

Click the component to open a new drill-down window. The window summarizes the information of the selected node along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill down on any field. Additionally, you can view the search results for the selected set of data by clicking **View Logs** in the same window.

Column

The Column Chart is a vertical bar graph that represents categorical data in rectangular bars with heights proportional to the values that they represent.

The Column Chart shows comparisons among discrete categories. It is a two-dimensional graph in which one axis of the graph shows the specific groups being compared and another one represents the measured value.

In the Single Aggregation with Grouping response type, the x-axis of a Column chart represents the values of the grouping parameter(s) whereas the y-axis represents the values of the aggregation parameter.

Example:

```
severity=* | chart count() by severity order by count() desc
limit 5
```

Line

The Line chart displays information as a series of data points called markers. The markers are connected to each other by a line.

The Line chart consists of two axes, in which x-axis contains the value of the grouping parameter(s) and the y-axis contains the values of the aggregation parameter. It is similar to a Column chart, except that, a Column chart usually displays discrete values, whereas a line chart visualizes a trend in continuous data.

Example:

```
severity=* | chart count() by severity
```

Donut Chart

The Donut Chart shows the data distribution based on the length of its arc. It was introduced in SLS to replace the Pie Chart. The reason for this is that pie charts can be hard to interpret as they focus on the proportional areas of the slices. Donut charts de-emphasize the use of area and focus on the lengths of the arcs of each individual element.

Example:

```
source_address=* | chart count() by source_address
```

Area Chart

The Area chart is used to represent quantitative data graphically. The graph is used to interpret the quantitative statistics graphically. The graph is based on a Line graph, and the area between the x-axis and lines are emphasized with colors, textures or hatchings.

Area charts are used to represent accumulated totals using numbers and percentages. It is also used to show the trends over time along with all related attributes.

Similar to the Line graphs, the x-axis of the Area chart represents the grouping parameter(s), and the y-axis represents values of the aggregation parameter.

Example:

```
action=* source_address=* | chart count() by action, source_
address
```

Bar Chart

The Bar chart is a horizontal bar graph that visualizes categorical data in a rectangular bar with the width proportional to the value.

In a Bar Chart, the x-axis represents the aggregation parameter and the y-axis represents the grouping parameter(s). Besides this, it is similar to the Column Chart.

Example:

```
severity=* | chart count() by severity order by count() desc
limit 5
```

Heatmap

Heatmap visualizes individual values in a matrix and represents them through different color shades based on their intensity. Use it to analyze the differences across multiple variables, reveal patterns, and detect correlations between them.

Example:

```
source_address=* action=* | chart count() by source_address,
action order by count() desc limit 10
```

Rendering Parameters

Click the settings icon at the top-right corner of the heatmap chart to open a dialog box. The dialog box allows you to configure the rendering parameters of the Heatmap chart.

The Rendering Parameters such as **X-axis Group**, **Positive Value**, and **Negative Value** provide a custom settings option to view data in different formats.

By default, the first grouping parameter of the query is assigned to the X-axis of the Heatmap. For example, the grouping parameter **source_name** is assigned in the X-axis of the Heatmap for the query:

```
| chart count() by source_name, action
```

However, by selecting a value from the drop-down menu of the **X-axis Group**, you can choose the grouping parameter to be placed on the X-axis of the chart. For example,

```
| chart count() by source_name, action
```

The query above contains two grouping parameters: **source_name** and **action**. If you choose **action** for x-axis, **source_name** is shown on y-axis. The **count()** value is represented according to the transparency level of the chosen cell color.

NOTE

If a query contains three or more than three grouping parameters, and you choose to keep **grouping_parameter_1** on the x-axis, then the combination of **grouping_parameter_2**, ……, **grouping_parameter_n** is shown on the y-axis.

Furthermore, you can assign custom colors to the Heatmap for both positive and negative values. SLS uses the selected color to represent the maximum value of the data obtained, and lesser values have the same color with linear transparency.

Radar

The Radar chart is a graphical representation of multivariate data in the form of a two-dimensional graph, in which one or more quantitative variables are represented on axes starting from the same point.

Each value of grouping parameter(s) forms an individual axis which is arranged radially around a point. These axes are equiangular to each other and known as spoke or radii. Each node depicts the value of a spoke, and the lines are drawn to connect the nodes to each other.

The Radar chart is best for visualizing outliers in a dataset, especially in cases of operation related analysis such as performance metrics and quality improvement. The line between the origin points and radii can be used as the scale for data points.

Example:

```
service=* action=* | chart count() by action, service
```

TreeMap

The Treemap chart is used to visualize the hierarchical structure of a tree diagram also displaying weight for each node via the area size. Each node is assigned to a rectangular area with their child nodes nested inside it. The space of each node inside a parent node is displayed with proportion to all other nodes within the same parent node. Also, the area size of the parent node is the total of its child nodes. If the weight of a children node is zero, then the node is not included in the treemap diagram.

The first grouping parameter is the parent node of a Treemap diagram, and all its successive parameters are the child nodes.

The name of the first grouping parameter is displayed in the breadcrumb, while all its fields are displayed in the containers as individual nodes.

Example:

```
source_address=* action=* | chart count() by source_address,
action order by count() desc limit 10
```

NOTE

The aggregation parameter determines the area size of each node in the container.

Rendering Parameters

Click the *gear* icon on the right side of the breadcrumb to select the rendering parameters for the nodes of the treemap chart.

You can choose one of the following type in the rendering parameters

- Single
- Unique
- Gradient

If the *Single* type is selected, all the nodes in the container are represented by a single color. You can also select the color to represent the nodes from the *Color* picker tool.

If the *Unique* type rendering parameter is selected, all the nodes in the container are represented by a unique color. The colors are chosen randomly by the SLS itself.

If you select the *Gradient* type rendering parameter, the **Color High** represents the node with the most significant area size and **Color Low** represents the node with the least area size.

Each section has a defined color, and different shades of color represent all the nodes of the division. The darkest shade represents the node with the most significant area size, and the shade of the color fades as the area size of the nodes decrease.

You can select the color for the nodes of high area value and low area value from *Color High* and *Color Low* drop-down menu respectively.

Operations

Zoom In and Zoom Out

The Zoom In feature allows you to click any node of the container and expand the chart further.

The expanded diagram displays the nodes of the successive grouping parameter associated with the selected parent node. The new node is shifted to the breadcrumb, and the container is updated with the fields of the node in the breadcrumb.

For example: In the diagram above, when the user clicks **source_address** `10.45.3.252`, it is shifted to the breadcrumb and all its related fields are displayed in the container.

With the Zoom Out feature, you can go back to the previous state of the diagram by clicking the breadcrumb.

Sankey

Sankey chart is a flow diagram used to depict a flow from one set of values to another. The connected values are called *nodes* and the connections are called *links*. It displays the corresponding grouping parameters on top of each node of the chart. The width of the link shows the magnitude of the flow. Colors are used to divide the diagram into different nodes or to show the transition from one state of the process to another.

Use the Sankey chart to show a *many to many* mapping between two or more nodes. The *aggregation parameter* is used to define the width of the flow between a source node and the destination node.

Example:

```
| process geoip(source_address) as country | chart count() by
country, severity, category, sub_category
```

Operations

Vertical Reposition

You can change the vertical position of the node(s) by dragging them in upward or downward direction. The nodes can either be overlapped or placed distinctly.

Parallel Coordinate

The Parallel Coordinate graph is a visualization technique used to plot individual data elements across multiple dimensions. The Parallel Coordinate graphs are ideal for comparing many grouping parameters and revealing the relationships between them. Each grouping parameter has its axis, and all the axes are placed in parallel to each other. Values are plotted as a series of lines that are connected across all the axes. This means that each line is a collection of points placed on each axis, which have all been linked together.

The Parallel Coordinates chart can show both the forest and the tree. The overall picture can be seen in the patterns of lines; individual lines can be highlighted to see the performance of specific value of the parameters. It is most useful in situations when the behavior of particular parameters may not be of concern, but combining them may emphasize an abnormal pattern or relationship.

Example:

```
| process geoip(source_address) as source_country | chart count()
by source_country, sub_category, destination_
location
```

Some notable points about the Parallel Coordinate chart are as follows:

1. Each line represents a relationship between two parameters rather than a trend or a change in value.

2. As the number of values increase, the graph may be cluttered or even overlapped at times, which makes it difficult to analyze. In this case, use the Brushing feature to highlight an individual or a group of values for better understanding.
3. You can view the value of the aggregation parameter by hovering over a relationship line.

Operations

Brushing

The Brushing feature eliminates one of the primary drawbacks of the Parallel Coordinate chart, which is cluttering and overlapping of the graph. When the number of data items in a Parallel Coordinate chart gets very high, lines get cluttered and even overplotted which eventually becomes difficult to understand. Using the brushing feature, you can select an area containing one or many data points.

The line(s) under the brushed area is highlighted. You can view the details of the stressed relationship by hovering over the particular line. In addition to that, you can further drill-down to its details of the relationship by clicking it.

Combined Drill-down

In addition to the regular drill down operation, you can also perform a combined drill-down using the **Brush**. Select a range of values in multiple axes using a brush and click the brushed area, to drill down.

The results of the drill-down filters down to the combination of the selected grouping parameter values.

Changing order of the parameter

By default, the first grouping parameter of the query is assigned to the first axis of the Parallel Coordinate chart, followed by the other grouping parameters.

For example, in the above query, the grouping parameter **source_address**, **sub_category** and **destination_location** are placed in first, second and third axes respectively.

You can also change the order of parameters by dragging them across the parameters with which the value is to be exchanged.

World Map

A World Map is a map of a country, continent, or region map, with colors and values assigned to specific regions. Values are displayed as a color scale, and you can view the name of the country by hovering over a particular part.

In Single Aggregation with Grouping, the color shade on each region of a World Map displays the value of the aggregation parameter, i.e., higher the value of the aggregation parameter, darker the color.

Some notable points about the World Map:

1. Sections of a graph are only clickable if they have some value of aggregation parameter and the search query contains two or three grouping parameter meaning that you cannot click and further drill-down the chart for a query with one grouping parameter or more than three grouping parameters.
2. For search queries with a single aggregation parameter and two grouping parameters, you can view a Donut chart by clicking on any region of a World Map (with some value for the aggregation parameter).
3. For search queries with a Single aggregation parameter and three grouping parameters, you can view a Heatmap by clicking on any region of a World Map(with some value for the aggregation parameter).

NOTE

You can drill-down operations from these sub-charts.

Rendering Parameters

Clicking the gear icon at the top right corner of the World Map opens a dialog box. The dialog box allows you to configure the rendering parameters of the World Map.

The Rendering Parameters such as **Country**, **Positive Value**, and **Negative Value** provide custom settings option to view data in different formats.

Through the **Country** option, you can specify the grouping parameter containing the names of the countries.

Whereas, the **Positive Value** and **Negative Value** options allow you to select the color to represent the positive values of the aggregation parameter and the negative value of the aggregation parameter respectively.

Operations

Pan and Zoom

The **Pan and Zoom** feature allows you to zoom in and out on a specific section on the world map and shift from one section to another.

ATT&CK

The **ATT&CK** chart is a heatmap describing the attacks carried out on a system in the form of the attack tactics and techniques described by MITRE. The chart is displayed if the grouping parameter contains the **attack_id** field.

To populate the ATT&CK chart, SLS adds the following fields to the corresponding logs each time an alert is triggered:

1. **attack_id**: An ID for the attack.
2. **attack_category**: The type of attack tactic used.
3. **attack_tag**: The type of attack technique used.

The header row of the ATT&CK chart contains the different tactics that may be used to perform an attack. The body of the chart contains the techniques used to execute the corresponding tactics. Finally, the heatmap is generated based on the frequency of the techniques.

Example:

```
| chart count() by attack_id
```

Multiple Aggregation without Grouping

The Multiple Aggregation without Grouping response type is used for aggregation of multiple aggregation parameters for all the available logs or the given repo and time range. An example of a search query for such response is:

```
| chart count(), avg(datasize)
```

This query displays the total count and the average of datasize of the logs collected in the specified range of time. The result of this query can be represented in the form of **Clustered Column, Clustered Bar** and **Display** charts.

General Operations for Multiple Aggregation without Grouping

Interactive Legend

In **Multiple Aggregation without Grouping**, when a chart is rendered, all selected aggregation parameters are displayed with a unique color for each aggregation parameter. However, you can choose to view the graphs concerning a specific aggregation parameter(s).

To hide an aggregation parameter, click the name of the parameter on the legend at the extreme right side of the container.

When you click a name of a parameter on the legend, the section referring to the respective parameter disappears, and a new chart is rendered consisting of all other aggregation parameters. You can unhide the parameter by clicking the legend again.

NOTE

The scale on the y-axis is auto-adjusted as per the value of the remaining aggregation parameter(s).

Drill down

Through the drill-down feature, you can choose to retrieve detailed results about a specific section of a chart. In Multiple Aggregation without Grouping response type, you can drill-down search operation over a specific value of aggregation parameter.

Hover over a component of a graph (example: node, line, bar, point) to view the tooltip. The tooltip displays all the relevant information about the particular component.

Click the segment to open a drill-down window. The window summarizes the related information of the selected section along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill-down the search result from any field. Additionally, click the **View Logs** to view the search result for the selected set of data.

NOTE

The drill-down feature is not applicable for the Display format of Multiple Aggregations without Grouping response type.

Clustered Column

The Clustered Column chart is a type of Column chart which allows you to display multiple quantitative variables.

Unlike a standard Column chart, where only one variable is used to mark x-axis, a Clustered Column chart uses multiple variables on the x-axis with a different color for each variable.

For the Multiple Aggregation without Grouping response type, the x-axis represents the different aggregation parameter, and y-axis contains the scale that denotes the value of the aggregation parameter.

Example:

```
| chart max(sent_datasize), max(received_datasize)
```

Clustered Bar chart

The Clustered Bar chart is a horizontal bar graph that represents multiple categorical data in a rectangular bar with the width proportional to the value.

The only difference between a Clustered Bar chart and a Clustered Column chart is the placement of parameters. In a Clustered Column chart, the aggregation parameter is placed on the x-axis whereas, in a Clustered Bar chart, the parameters are placed in the y-axis.

Example:

```
| chart avg(sent_datasize), avg(received_datasize)
```

Display

The **Display** format shows the value of an aggregation parameter in the container.

For Multiple Aggregation without Grouping response type, the value of the first aggregation parameter is displayed in the container.

To view the search results in display format, select *Display* from the drop-down menu on the top right corner of the Search result page.

Rendering Parameters

Click the gear icon at the top right corner of the display container to configure the rendering parameters. You can configure the output format, color, and layout in the panel.

You can choose the output format, font and the background color from the rendering parameter section.

> **NOTE**
>
> The **Output Format** section is disabled when you select the default layout.

Multiple Aggregation with Grouping

The Multiple Aggregation with Grouping response type is used for aggregation of grouping parameters concerning given multiple aggregation parameters.

The general syntax for **Multiple Aggregation with Grouping** is:

```
| chart aggregation_parameter1, aggregation_parameter2 by
grouping_parameter1, grouping_parameter2, ...,
grouping_parametern
```

Example queries of Multiple Aggregation with Grouping type are:

```
| chart count(), avg(datasize) by action

user=* | chart count(label=Fail) as Failed, count
(label=Successful) as Successful by user order by Failed desc
limit 10
```

This query displays the count and average datasize of the collected logs in the specified time range grouped by the actions applied. The result of this query can be represented in the form of **Clustered Column**, **Clustered Bar**, **Clustered Line**, **Stacked Area**, **Radar**, **World Map**, and **Bubble** charts.

General Operations for Multiple Aggregation with Grouping

This section contains the general operations that can be applied to all the charts belonging to the Multiple Aggregation with Grouping response type.

NOTE

Some charts might consist of operations that are relevant to the specific chart only. For such operations, refer to the section of the particular chart.

Interactive Legend

In the Multiple Aggregation with Grouping response type, when a chart is rendered, values of all the selected aggregation parameters are displayed with a unique color for each aggregation parameter. However, you can choose to view the graphs concerning specific aggregation parameter(s).

To hide an aggregation parameter, click the name of the parameter on the legend at the extreme right side of the container.

Click the name of a parameter on the legend, to hide its respective section. A new chart is rendered consisting of all other aggregation parameters. You can unhide the parameter by clicking the legend again.

NOTE

The scale on the y-axis is auto-adjusted as per value of the remaining aggregation parameter(s).

Drill-down

In the Multiple Aggregation with Grouping response type, you can drill-down search operation regarding a specific value of the grouping parameter concerning a single or multiple aggregation parameters.

Hover over a component of a graph (example: node, line, bar, point e.t.c) to view a tooltip. The tooltip displays all the relevant information about the particular component.

Click the segment to open a drill-down window. The window summarizes the related information of the selected section along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill-down the search result from any field. Additionally, click the **View Logs** to view the search result for the selected set of data.

Clustered Column

The Clustered Column chart is a type of Column chart in which you can display multiple quantitative variables.

Unlike standard Column chart in which only one variable is used to mark the x-axis, the Clustered Column chart uses multiple variables on the x-axis with a different color for each variable.

For the Multiple Aggregation with Grouping response type, the x-axis contains the values of grouping parameter(s) with a vertical bar for each aggregation parameter. The height of the bar determines the value of the aggregation parameter for the specific value of a grouping parameter. The y-axis contains the scale that denotes the value of the aggregation parameter.

Example:

```
action=Allow or action=Deny | chart count(action=allow) as
AllowedConnection, count(action=
deny) as DeniedConnection by source_address order by count
(action=allow), count(action=
deny) desc limit 10
```

Clustered Bar

The Clustered Bar chart is a horizontal bar graph that presents multiple categorical data in a rectangular bar with the width proportional to the value.

The only difference between a Clustered Bar chart and Clustered Column chart is the placement of parameters. In a Clustered Column chart, the grouping parameter is placed on the x-axis whereas, in a Clustered Bar chart, the grouping parameters are placed on the y-axis.

Example:

```
action=Allow or action=Deny | chart count(action=allow) as
AllowedConnection, count(action=
deny) as DeniedConnection by source_address order by count
(action=allow), count(action=
deny) desc limit 10
```

Clustered Line

The Clustered Line chart is an extension of the line graph in which multiple lines are used to represent values of different categories. Silimar to the Clustered Column chart, in a Clustered Line chart the x-axis contains the values of the grouping parameters, and the y-axis contains the scale to measure the value of an aggregation parameter of the particular grouping parameter.

Example:

```
sent_datasize=* source_address=* | chart max(sent_datasize), max
(received_datasize) by source_
address order by max(sent_datasize), max(received_datasize) desc
limit 10
```

Stacked Area

Stacked Area charts are fundamentally similar to a standard Area chart, except for the use of multiple variables in the x-axis instead of a single variable.

In the Multiple Aggregation with Grouping response type, the x-axis contains the values of the grouping parameter(s), whereas the y-axis consists the scale to measure the value of the aggregation parameters. A unique color represents each aggregation parameter.

```
sent_datasize=* source_address=* | chart max(sent_datasize), max
(received_datasize) by source_
address order by max(sent_datasize), max(received_datasize) desc
limit 10
```

Radar

The Radar chart is a graphical representation of multivariate data in two-dimensional space. The chart is used to visualize the outliers in the dataset, especially in cases of operation related analysis such as performance metrics and quality improvement.

To know more about Radar charts refer to the Radar chart section in Single Aggregation with Grouping response type.

Example:

```
"norm_id"="WinDNSDHCP"| chart count(lease_address=end), count
(lease_address=start) by user
```

World Map

A World Map is a map of a country, continent, or region with colors and values assigned to specific regions. Values are displayed as a color scale, and you can view the name of the country by hovering over a particular part.

In Multiple Aggregation with Grouping, the color shade on each region of a World Map displays the value of the first aggregation parameter, i.e., higher the value of the aggregation parameter, darker the color. The values of all other successive aggregation parameters can be viewed using the sub-charts.

Example:

```
| process geoip(destination_address) as country_name | chart
count(), avg(datasize) by country_name, action
```

NOTE

1. The value of the first grouping parameter can only be depicted in the chart if the search query contains only one grouping parameter. However, the value can be viewed from the **Search Table**.
2. Sections of a graph are only clickable if they have some value of aggregation parameter and the search query contains two or three grouping parameters. Thus, you cannot click and further drill-down the chart for a query with one grouping parameter or more than three grouping parameters.
3. For search queries with multiple aggregation parameters and two or three grouping parameters, you can view a Clustered Column chart by clicking on any region of a World Map (with some value for the aggregation parameter).

NOTE

You can further drill down from these charts.

Rendering Parameters

Click the gear icon at the top right corner of the World Map to open the rendering parameters panel.

The Rendering Parameters such as **Country**, **Positive Value**, and **Negative Value** provide a custom settings option to view data in different formats.

Through the **Country** option, you can specify the grouping parameter containing the names of the countries. Whereas, the **Positive Value** and **Negative Value** options allow you to select the color to represent the positive values of the aggregation parameter and the negative value of the aggregation parameter respectively.

Operations

Pan and Zoom

The **Pan and Zoom** feature allows you to zoom in and out on a specific section on the world map and shift from one section to another.

Bubble Chart

The Bubble Chart is a scatter chart that shows the relationship between variables using three dimensions: the x-axis, the y-axis, and the bubble radius. The chart can display different groups of data at once. Based on the grouping parameter, the chart groups the data into bubbles of different colors with each color representing a single group.

You can see the group names and their corresponding colors in the legend to the right of the chart.

Example:

```
| chart count(), max(sig_id) by action
```

By default, in the search command for the Bubble Chart, the first aggregation parameter represents the x-axis while the next two parameters represent the y-axis and the bubble radius respectively.

You can also use the Bubble Chart with more than three aggregation parameters. To see the values of the other parameters, hover over a bubble in the chart. A tooltip appears, displaying all the values of the parameters associated with the bubble chart.

NOTE

The radii with negative values are represented in the red-colored text. However, the system takes the modulus of the negative value and plots it in the chart.

Example:

```
| chart count(), max(sig_id), distinct_count(action),
distinct_count(sig_id) by action
```

Rendering Parameters

Click the settings icon at the top-right corner of the Bubble Chart to open a dialog box. The dialog box allows you to configure the rendering parameters of the Bubble Chart.

You can select the required parameters from the **Plot in Y-axis** and **Plot at Radius** drop-down menus to represent the y-axis and the bubble radius respectively.

NOTE

Make sure you select different parameters to represent the y-axis and the bubble radius.

Timechart Single Aggregation without Grouping

The Timechart Single Aggregation without Grouping response type is used for aggregation of processed logs to a given aggregation parameter grouped into time buckets (as a time series data) over a specified time range.

The general syntax for the **Timechart Single Aggregation without Grouping** is:

```
| timechart aggregation_parameter
```

Example queries of the Timechart Single Aggregation without Grouping type are:

```
| timechart count()
```

```
| timechart sum(datasize)
```

```
| timechart avg(datasize)
```

This response type displays the value of the aggregation parameter in the specified range of time. The **Column**, **Line**, **Area**, **Day/Hour Heatmap**, and **Radar** charts are used to visualize the queries belonging to this response type.

Additionally, the **Cumulative** chart option is also available along with the **Normal** chart for the **Column**, **Line**, and **Area** charts. The **Cumulative** option visualizes the results by accumulating data from the starting point to the current time-bucket for all time-buckets whereas the normal option visualizes the results as obtained from the query.

General Operations for Timechart Single Aggregation without Grouping

Drill-down

You can choose to view a detailed search for the response type regarding a specific value in two ways, i.e., from the line, or using a drag box.

Hover over a specific component/area of a chart to view a tool-tip. The tooltip displays all the information about the particular node.

Click the component to open a drill-down window. The window summarizes the related information of the selected section along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill-down the search result from any field. Additionally, click the **View Logs** to view the search result for the selected set of data.

In addition to that, you can also drill down any chart of the response type using the drag box. Click and drag the mouse inside the graph, a yellow colored transparent drag box appears. You can drill-down the selected section of the chart by clicking the drill-down icon on the top-right corner of the box. You can resize or move the drag box as per your requirement.

Cumulative chart

The Cumulative chart displays the accumulated data values throughout the given time range. To view the cumulative chart, click *Cumulative* on the left side of the container of a chart.

Click *Normal* to view the regular chart.

Trendline

You can select the *Show/Hide Trendline* checkbox to identify whether the time-series data is likely to increase, decrease, or remain constant over a time period. The data on an increasing trend forms an upsloping line, whereas, on a decreasing trend, it forms a downsloping line. The *Show/Hide Trendline* checkbox is available for **Column**, **Line**, and **Area** charts of this response type only.

> **NOTE**
>
> The *Show/Hide Trendline* checkbox is also available for **Column**, **Line**, and **Area** charts resulted from Simple search queries and a blank search query.

Interactive Animation

The charts belonging to the **Timechart** response type include an interactive play button. The button allows you to slide through values of the charts concerning time buckets known as **Interval**.

Click **Play** on the right side of the container to start the animation. The graph is refreshed every four seconds, i.e., that graph shifts from one time-bucket to another time bucket every four seconds. Value of the time bucket is dependent upon the time-range specified in the **Interval**.

You can also click *Pause*, *Stop*, *Previous*, *Next*, *Replay* as required.

> **NOTE**
>
> The operations **Cumulative chart** and **Interactive Animation** are not available for the Radar chart.

Column

The Column chart is a vertical bar graph that presents categorical data in rectangular bars with heights proportional to the values that they represent.

The Column chart shows comparisons among discrete categories. It is a two-dimensional chart in which, one axis of the chart shows the specific categories being compared and another one represents the measured value.

In the Timechart Single Aggregation without Grouping response type, the x-axis of the Column chart represents the value of timestamps whereas the y-axis represents the values of the aggregation parameter.

Each bar represents the value of the aggregation parameter in a given *Interval*. The *Interval* is calculated automatically as per the time range selected in the *Search Bar . The value of the *Interval* is displayed on the extreme left of the container.

Example:

```
| timechart avg(datasize)
```

Line

The Line chart or line graph displays information as a series of data points called markers, connected to each other by a line segment. It is similar to a Column chart, except a Column chart usually displays discrete values, whereas the Line chart visualizes a trend in continuous data.

Similar to the Column chart, the line chart also consists of two axes, in which the x-axis contains the value of timestamps and the y-axis contains the values of the aggregation parameter.

Example:

```
| timechart avg(datasize)
```

Area

The Area chart is used to represent quantitative data graphically. The graph is based on the Line chart, and the area between the axis and lines are emphasized with colors, textures or hatchings.

Area charts are used to represent accumulated totals using numbers and percentages. It is also used to show the trends over time along with all related attributes.

Similar to the line charts, the x-axis of the area chart represents the value of timestamps whereas the y-axis represents the values of the aggregation parameter.

Example:

```
| timechart sum(datasize)
```

Radar

The Radar chart is a graphical representation of multivariate data in two-dimensional space. The chart is used to visualize the outliers in the dataset, especially in cases of operation related analysis such as performance metrics and quality improvement.

To know more about Radar charts refer to the Radar chart section in Single Aggregation with Grouping response type.

The Radar chart can be used in time queries to graphically represent the change in values of the aggregation parameter over a period.

Example:

```
"norm_id"="WinDNSDHCP"| timechart count(lease_address=drop)
```

Day/Hour Heatmap

Heatmap is used to visualize individual values contained in a matrix and represent them using different shades of a single color as per their intensity.

The Day/Hour Heatmap is an extension of a regular heatmap in which results are displayed in the day/hour format. The Day/Hour Heatmap has seven rows and 24 columns, each row representing a day of the week and columns representing hours of a day. That means each cell represents a specific hour of a particular day of a week. The Day/Hour Heatmap only works for the Timechart Single Aggregation with Grouping response type with **every 1 hour** suffixed to the query.

The query format for the Day/Hour Heatmap is:

```
| timechart aggregation_parameter1 every 1 hour
```

The values of the aggregation parameter are displayed in the cells as per their timestamps.

The intensity of the color is dependent upon the relative value of the aggregation parameters.

Example:

```
| timechart sum(datasize) as TotalDatasize every 1 hour
```

When the selected time range is more than a week, a slider appears on the right end of the container that allows the user to slide over the particular days.

Rendering Parameters

You can assign custom colors to the Day/Hour heatmap for both positive and negative values. SLS uses the selected color to represent the maximum value of the data obtained, and lesser values have the same color with linear transparency.

Timechart Single Aggregation with Grouping

The Timechart Single Aggregation with Grouping response type is used for aggregation of processed logs by an individual grouping parameter concerning given a single aggregation parameter grouped into time buckets (as a time series data) over a specified time range.

The general syntax for **Timechart Single Aggregation with Grouping** is:

```
| timechart aggregation_parameter by grouping_parameter1,
grouping_parameter2, ....., grouping_parametern
```

Example queries of Timechart Single Aggregation with Grouping type are:

```
| timechart count() by action
```

This query displays the count of the logs generated by the individual action, for an individual time bucket over a specified range of time. The result of this query can be represented in the form of **Clustered Line** and **Stacked Column** charts.

General Operations for Timechart Single Aggregation with Grouping

This section contains the general operations that can be applied to all the charts belonging to the Timechart Single Aggregation with Grouping response type.

> **NOTE**
>
> Some charts might consist of operations that are relevant to the specific chart only. In this case, refer to the section of the particular chart.

Interactive Legend

In the Timechart Single Aggregation with Grouping response type, when a chart is rendered, all the aggregation values of the selected grouping parameter(s) are displayed with a unique color for each value of the grouping parameter(s). However, you can choose to view the graphs concerning a specific value of grouping parameter(s).

To hide the value of a grouping parameter, click the name of the parameter on the legend at the extreme right side of the container.

When you click a name of a parameter on the legend, the section (line, bar) referring to the respective parameter disappears, and a new chart is rendered consisting all other values of the grouping parameter(s). Click the legend again to unhide the particular value.

> **NOTE**
>
> The scale on the y-axis is auto-adjusted as per the value of the remaining values of grouping parameter(s).

Drill-down

You can choose to view a detailed search for the response type regarding a specific value in two ways, i.e., from the line, or using a drag box.

Hover over a specific component/area of a chart to view a tool-tip. The tooltip displays all the information about the particular node.

Click the component to open a drill-down window. The window summarizes the related information of the selected section along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill-down the search result from any field. Additionally, click the **View Logs** to view the search result for the selected set of data.

In addition to that, you can also drill-down any chart of the response type using the drag box. Click and drag the mouse inside the graph, a yellow colored transparent drag box appears. You can drill-down the selected section of the chart by clicking the drill-down icon on the top-right corner of the box. You can resize or move the drag box as per your requirement.

Interactive Animation

The charts belonging to the **Timechart** response type include an interactive Play button. It allows you to slide through values of the charts concerning time buckets known as **Interval**.

Click the **Play** on the right side of the container to start the animation. The graph is refreshed every four seconds, i.e., that graph shifts from one time-bucket to another time bucket every

four seconds. Value of the time bucket is dependent upon the time-range specified in the **Interval**.

You can also click *Pause*, *Stop*, *Previous*, *Next*, *Replay* as required.

Clustered Line

The Clustered Line graph is an extension of the Line chart in which multiple lines are used to represent values of different categories or different values of a category.

In the Timechart Single Aggregation with Grouping, the y-axis represents the aggregation value for every grouping parameter, and the x-axis displays the value of the timestamps. Similarly, the lines represent the values of the grouping parameter(s).

Example:

```
event_category=* | timechart count() by event_category
```

Stacked Column

A Stacked Column chart uses bars to show the comparisons between categories of data but with an ability to break down and compare parts of a whole. Each bar in the chart represents a whole, and segments in the bar represent different parts or categories of that whole.

Similar to the Clustered Line chart, the y-axis represents value of the aggregation parameter, and the x-axis displays value of the timestamps.

The single bar represents a time-bucket whereas the segments of the bar are used to represent values of the grouping parameter. Each value of the grouping parameter has a unique color, and the small segment is stacked upon one another to form a complete bar. The vertical length of a segment indicates its value of the aggregation parameter.

Example:

```
source_address=* | timechart count() by source_address
```

NOTE

If the search result contains a large number of data points (more than 50) or groups (more than 20), switching from the Clustered Line to Stacked Column consumes a large amount of CPU resources. In this case, SLS displays the following message.

Timechart Multiple Aggregation without Grouping

The Timechart Multiple Aggregation without Grouping response type is used for aggregation of processed logs related to the given parameters. The logs are grouped into time buckets (as a time series data) over a specified time-range.

The general syntax for **Timechart Multiple Aggregation without Grouping** is:

```
| timechart aggregation_parameter1, aggregation_parameter2, ....
aggregation_parametern
```

Example queries of Timechart Multiple Aggregation without Grouping type are:

```
| timechart count(), avg(datasize)
```

This query displays the count of total logs generated and the average datasize of collected logs for individual time bucket over a specified range of time. The result of this query can be represented in the form of **Clustered Column**, **Clustered Line**, **Radar**, and **Stacked Area** charts.

General Operations of Timechart Multiple Aggregation without Grouping

This section contains the general operations that can be applied to all the charts belonging to the Timechart Multiple Aggregation without Grouping response type.

> **NOTE**
>
> Some charts might consist of operations that are relevant to the specific chart only. For such operations, refer to the section of the particular chart.

Interactive Legend

In the Timechart Multiple Aggregation without Grouping response type, when a chart is rendered, all the values of the selected aggregation parameter(s) are displayed with a unique color for each value of the aggregation parameter(s). However, you can view the graphs for specific aggregation parameter(s).

To hide an aggregation parameter, click the name of the parameter on the legend at the extreme right side of the container.

When you click a name of a parameter on the legend, the section (line, column, bar) referring to the respective parameter disappears, and a new chart is rendered consisting all other aggregation parameters (s). Click the legend again to unhide the value.

> **NOTE**
>
> The scale on the y-axis is auto-adjusted as per the value of the remaining aggregation parameter(s).

Drill-down

You can choose to view a detailed search for the response type regarding a specific value in two ways, i.e., from the line, or using a drag box.

Hover over a specific component/area of a chart to view a tool-tip. The tooltip displays all the information about the particular node.

Click the component to open a drill-down window. The window summarizes the related information of the selected section along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill-down the search result from any field. Additionally, click the **View Logs** to view the search result for the selected set of data.

In addition to that, you can also drill-down any chart of the response type using the drag box. Click and drag the mouse inside the graph, a yellow colored transparent drag box appears. You can drill-down the selected section of the chart by clicking the drill-down icon at the top-right corner of the box. You can resize or move the drag box as per your requirement.

Interactive Animation

The charts belonging to the **Timechart** response type include an interactive play button. It allows you to slide through values of the charts concerning time buckets known as **Interval**.

Click **Play** on the right side of the container to start the animation. The graph is refreshed every four seconds, i.e., that graph shifts from one time-bucket to another time bucket every four seconds. Value of the time bucket is dependent upon the time-range specified in the **Interval**.

You can also click *Pause*, *Stop*, *Previous*, *Next*, *Replay* as required.

Clustered Column

The Clustered Column chart is a type of a Column chart in which you can display multiple quantitative variables.

Unlike a standard Column chart, the Clustered Column chart uses multiple variables on the x-axis with a different color for each variable.

For the Timechart Multiple Aggregation without Grouping response type, the x-axis represents the different time buckets within the specified time range, and the y-axis contains the scale that denotes the value of the aggregation parameter. The bars indicate the different values of the aggregation parameter at different timestamps. The vertical length of a bar signifies its value of the aggregation parameter at that particular timestamp.

Example:

```
norm_id=WinDNSDHCP | timechart count(lease_address=drop) as
Dropped, count(lease_address=start) as Started,
count(lease_address=end) as ENDED
```

Clustered Line

The Clustered Line graph is an extension of a Line chart in which multiple lines are used to represent values of different categories or different values of a category.

Alike to the Clustered Column chart, the y-axis represents values of the aggregation parameter, and the x-axis displays the value of the timestamps. Similarly, the lines represent the values of the aggregation parameters at a particular timestamp.

Example:

```
| timechart count("event_category" = "THREAT") as Dangerous,
count("event_category" ="TRAFFIC") as Traffic
```

Radar

The Radar chart is a graphical representation of multivariate data in two-dimensional space. The chart is used to visualize the outliers in the dataset, especially in cases of operation related analysis such as performance metrics and quality improvement.

To know more about Radar charts refer to the Radar chart section in Single Aggregation with Grouping response type.

The Radar chart can be used in time queries to graphically represent the change in values of aggregation over a period. For Timechart Multiple Aggregation without grouping type, each aggregation parameter is represented by a unique color.

Example:

```
norm_id=WinDNSDHCP | timechart count(lease_address=drop) as
Dropped, count(lease_address=start) as Started,
count(lease_address=end) as ENDED
```

Timechart Multiple Aggregation with Grouping

The Timechart Multiple Aggregation with Grouping response type is used for aggregation of an individual grouping parameter for given multiple aggregation parameters grouped into time buckets over a specified time range.

The general syntax for **Timechart Multiple Aggregation without Grouping** is:

```
| timechart aggregation_parameter1, aggregation_parameter2, ....,
aggregation_parametern by grouping_
parameter1, grouping_parameter2, ...., grouping_parametern
```

An example of a search query for the response is:

```
"norm_id"="WinDNSDHCP"  | timechart count("description" =
"THREAT") as Dangerous,  count("description" =
"TRAFFIC") as Traffic by lease_address
```

The result of this query can be represented in the form of **Clustered Column** and **Bubble** charts.

General operations for Timechart Multiple Aggregation with Grouping

Drill-down

Like in the search results of other responses, when you hover on any section (here, any count () or avg(doable_mps)), the selected section is highlighted, and the information for the selected section is as shown in the tooltip.

Click the component to open a drill-down window. The window summarizes the related information of the selected section along with the option to drill down as per your preference.

Click the corresponding **Open in a new window** icon to further drill-down the search result from any field. Additionally, click the **View Logs** to view the search result for the selected set of data.

In addition to that, you can also drill-down any chart of the response type using the drag box. Click and drag the mouse inside the graph, a yellow colored transparent drag box appears. You can drill-down the selected section of the chart by clicking the drill-down icon on the top-right corner of the box. You can resize or move the drag box as per your requirement.

Interactive Legend

For the responses of **Timechart Multiple Aggregation with Grouping**, the legend is displayed on either side of the search graph. The aggregation parameter(s) is shown on the left-hand side whereas the grouping parameter is shown on the right-hand side.

An important thing to note here is that at an instant, the result of only one of the grouping parameters is displayed. Moreover, only the legends of the grouping parameter (on the right) are interactive. The legends of aggregation parameters (on the left) are not interactive.

Interactive Animation

The charts belonging to the **Timechart** response type include an interactive Play button. It allows you to slide through values of the charts concerning time buckets known as **Interval**.

Click the **Play** on the right side of the container to start the animation. The graph is refreshed every four seconds, i.e., that graph shifts from one time-bucket to another time bucket every four seconds. Value of the time bucket is dependent upon the time-range specified in the **Interval**.

You can also click *Pause*, *Stop*, *Previous*, *Next*, *Replay* as required.

Clustered Column

The Clustered Column chart is a type of a Column chart in which you can display multiple quantitative variables.

Unlike a standard Column chart, the Clustered Column chart uses multiple variables on the x-axis with a different color for each variable.

For the Timechart Multiple Aggregation without Grouping response type, the x-axis represents the different time buckets within the specified time range, and the y-axis contains the scale that denotes the value of the aggregation parameter. The bars indicate the different values of

the aggregation parameter at different timestamps. The vertical length of a bar signifies its value of the aggregation parameter at that particular timestamp.

Example:

```
norm_id=WinDNSDHCP | timechart count(lease_address=drop) as
Dropped, count(lease_address=start) as Started,
count(lease_address=end) as ENDED
```

Bubble Chart

The Bubble Chart is a scatter chart that shows the relationship between variables using three dimensions: the x-axis, the y-axis, and the bubble chart can display different groups of data at once. Based on the grouping parameter, the chart groups the data into bubbles of different colors with each color representing a single group.

You can see the group names and their corresponding colors in the legend to the right of the chart.

Example:

```
| timechart count(), avg(sig_id) by status_code
```

By default, in the search command for the Bubble Chart, the timechart represents the **Time** paramter in the x-axis while the next two parameters represent the y-axis and the bubble radius respectively.

You can also use the Bubble Chart with more than three aggregation parameters. To see the values of the other parameters, hover over a bubble in the chart. A tooltip appears, displaying all the values of the parameters associated with the bubble.

NOTE

The radii with negative values are represented in the red-colored text. However, the system takes the modulus of the negative value and plots it in the chart.

Example:

```
| timechart count(), avg(sig_id), max(datasize),
distinct_count(sig_id) by status_code
```

Rendering Parameters

Click the settings icon at the top-right corner of the Bubble Chart to open a dialog box. The dialog box allows you to configure the rendering parameters of the Bubble Chart.

You can select the required parameters from the **Plot in Y-axis** and **Plot in Radius** drop-down menus to represent the y-axis and the bubble radius respectively.

NOTE

Make sure you select different parameters to represent the y-axis and the bubble radius.

3.6 Interesting Fields

The **Interesting Fields** are the relevant fields presented on the basis of the data distribution for the following types of queries:

- Simple Search (except the Table and Time Functions commands)
- One-to-One Commands
- Filtering Commands (except the latest command and the search command)

The **Interesting Fields** window appears at the bottom-left side of the search page after performing a search operation and displays the top 15 fields, sorted according to the occurrence of the fields in the search result.

The parameters used to measure the data distribution are:

- Percentage (%) - Displays the percentage of occurrences of a given field.
- Count (# of values) - Displays the number of unique values for the given field.
- Mean Deviation - Displays the value of the deviation from the average occurrence of the given fields.
- Median Deviation - Displays the value of the deviation from the fields with the highest number of similar occurrences.

3.6.1 Actions in the Interesting Fields

Sorting

You can sort the Interesting Fields either by clicking on the field header or the parameter header.

View Details

You can view the details of the Interesting Fields by clicking on the desired field from the list. A pop-up panel appears, containing the total number of occurrences of the selected field and its **Top 10** values with their distinct counts and percentages.

3.6.2 Adding Interesting Fields

You can add fields to the **Interesting Fields** window in the following ways:

Selecting the fields

The **All Fields** panel pops up once you click *Select Fields*. The panel lists the top 100 fields from the search results. You can select the desired fields from the list and the relevant parameter from the *Parameter to display* drop-down menu to enlist them in the **Interesting Fields** window.

Adding the fields

Click **Add Fields** to enter the names of the desired fields. These fields are added in the **Interesting Fields** window and in the **All Fields** panel regardless of their occurrence in the search result. You can view these fields in a different colored text in the **Interesting Fields** window.

NOTE

- You can add a maximum of 20 fields in the **Add Fields** panel.
- You can neither use special characters nor Unicode characters in the **Add Fields** panel.

Adding the fields from the search result drop-down

You can select the **Add this field to interesting fields** option from the drop-down menu on the key-value pairs to add the required field in the **Interesting Fields** window. The fields added from here are appended in the **Add Fields** panel.

NOTE

- The Percentage (%) parameter is displayed in the **Interesting Fields** window by default. If the percentage of a field is less than *0.005*, it is displayed as *0*.
- The **Interesting Fields** feature is enabled by default. You can disable **Interesting Fields** by selecting the **Disable Interesting Fields in Search Page** option under `My Preferences >> Search`. You can also collapse or expand the **Interesting Fields** window by clicking the window header.
- The **Interesting Fields** window is disabled if:
  - The **Data Privacy Module** is enabled in your system.
  - The queries **Aggregators**, **Pattern Finding**, **Table**, **Time Functions**, **search**, or **latest** are used.
- The values of the parameters in the **Interesting Fields** window are approximated if:
  - The number of fields in the **Interesting Fields** window is more than 100.
  - The distinct count of the given field is more than 100.
- The fields *log_ts*, *col_ts*, *msg*, *SLS_name*, *repo_name*, and *label* are not supported in the **Interesting Fields** window.
- The **Loading Interesting Fields** icon appears in the search result tool bar if the system takes time to load the values in the **Interesting Fields** window. The icon disappears once the values are completely loaded.

IMPORTANT

SLS does not compute the values of the Interesting Fields if you have hidden the **Histogram** and collapsed the **Interesting Fields** window.

3.7 Customizable Drilldown from Search Visualization

SLS provides a number of options for search result visualization. While visualizing the search results or the content of a widget, it is possible to dive deeper into the results by clicking the graphical representation. For example, while viewing a search result which includes the fields such as *destination_address*, *destination_port*, *source_address*, and *source_port* in the search query, it is possible to drill down to the results based on these parameters. Use the keys from the original query to drill down.

3.7.1 Common Features of Drill-down

Depending on the original query chosen to drill down from, the contents in the drill-down context menu varies. There are 3 types of drill-down options in SLS visualization:

1. Filter
2. Drilldown by

3. Top 10 drilldown by

The *Filter* type drill-down searches on the *Range*, the *Field*, and the *count()*. The *Drilldown by* and the *Top 10 drilldown by* types drill down on the *fields* and the *labels* respectively.

For example:

```
destination_address=* source_port=* destination_port=* source_
address=*
```

While performing drill-down from this query, the following context menu appears on the screen. It lists all three possible sections in a drill-down context menu.

1. **Filter**
   This section contains the following components depending on the original query:
   - **Range**: Displays the subset of the time-period from which you have chosen to drill-down. It is only displayed for queries containing the timechart command or logs plotted in a time series manner.
   - **count()**: Total number of logs.
   - **View Logs**: Lets you view the drilled-down logs. You can view them in the same or a new window by clicking **View Logs** in the context menu for the given time-range.

   > **NOTE**
   > By default, the **Drilldown on full result set** slider and **count()** are disabled (grayed out).

2. **Drilldown by**
   This section contains the fields or labels present in the original query.
3. **Top 10 Drilldown by**
   This section contains the fields or labels present in the original query.

Besides these, the context menu also contains some other options for the following.

- **Drilldown on Full Result Set**
  It is possible to drill down on the full result. The slider icon present next to the *Range* value lets you drill down on the full result set in addition to the subset.
- **Open drilldown in a New Window**
  While performing drill-down, it executes in the same window by default. However, you can click the **Open in New Window** icon to open the results in the new window.

3.7.2 Demonstration of Customizable Drilldown from Search Visualization

Consider the following search query:

```
device_ip=* device_name=* col_type=* source_address = 10.94.2.94
```

This query displays the following visualization.

SLS's search result drill-down actions let you dive deeper into the details of the information presented in the visualization. If you hover over the search graph, the related information of the selected area is summarized in a tooltip.

Click the highlighted section of the result.

In the context menu, enable or disable the drill-down on the *Range* value by clicking the slider icon. The corresponding search visualization for the *Range* is shown below:

Click **View Logs** to see the corresponding log results. The results can be viewed in the same window or in a new one.

Click the required *Field-values* in the **Drilldown by** section to see the corresponding search results. The results can be viewed in the same window or in a new one.

Click the *device_ip* in the *Drilldown by* section to append **chart count() by device_ip order by count() desc** in the search query. The search result can be viewed in the same window or in a new one.

```
device_ip=* device_name=* col_type=*  source_address = 10.94.2.94
| chart count() by device_
ip order by count() desc
```

Click the required *labels* in the **Top 10 drilldown by** section to see the corresponding search results. These results can be viewed in the same window or in a new one.

Click *device_ip* in the *Top 10 Drilldown by* section to append **| chart count() by device_ip order by count() limit 10 desc** to the search query. Choosing *device_ip* results in the following query.

```
device_ip=* device_name=* col_type=*  source_address = 10.94.2.94
| chart count() by device_
ip order by count() desc limit 10
```

Similarly, the search results can be drilled down on the basis of the *source_port*, *destination_port*, and the *source_address*.

The search result can be further drilled down by clicking any part of the result set.

```
device_ip=127.0.0.1 device_name=* col_type=*  source_address =
10.94.2.94 | chart count() by device_
name order by count() desc
```

3.7.3 Special Drilldown Scenarios

Filter Drilldown

Example 1

For **Filter Drilldown**, if you drill down on the *Range* and open the results in the same page, the search is executed in the selected time-range. If you open the search in a new window, it is executed in the selected time-range with **| timechart count()** appended to it. The command is appended only for simple queries.

Select a bar to drill down from. The following context menu appears.

Once you drill down, you can see results similar to the following example.

For the filter type, when the drill-down is executed on *Field*, search is executed with **| filter <field> = <value>**

Consider the following query:

```
action=*|chart count() by action
```

The following visualization appears.

If you drill down on the *reporting_Speed*, the following context menu appears.

If you drill down on the *reporting speed*, the appended search query is:

```
action=* | chart count() by action | filter "action"="reporting
speed"
```

Example 2

When the drill-down is executed on *count()* for the *Filter* type, the search is executed with **| search count() = <value>**. Consider the following example:

```
action=* | chart count() by action
```

The following visualization appears.

The context menu for this drilldown is:

When the drill-down is executed on *count(): 544*, the new appended query is:

```
action=*|chart count() by action | filter "count()" = 544
```

Example 3

When the drill-down is conducted for **<empty query> | chart count() by group**, the customizable drill-down options differ. Consider the following:

```
| chart count() by action
```

The following visualization appears.

Clicking drill-down for a bar opens up the following context menu. In this case, only the Filter section with *field, count()* and *View Logs* is displayed as shown.

If you click “action: reporting speed”, the new query becomes:

```
| chart count() by action | filter "action"="reporting speed"
```

If you click “count(): 544”, the new query becomes:

```
| chart count() by action | filter "count()"=544
```

Drilldown by

For **Drilldown by**, when the drill-down is executed on *fields* or *label*, search is executed with the given query followed by **| chart count() by <field> order by count() desc**

For example:

```
action = denied
```

The following visualization appears. Hover over the required result and click to drill down.

In the *Drilldown Context Menu*, click **action** under the Drilldown by section.

The search results of the drilldown appear.

New query:

```
action = denied| chart count() by action order by count() desc
```

Top 10 Drilldown by

For **Top 10 Drilldown by**, when you execute the drill-down on *field-values* or *label*, the search is executed with the given query followed by **| chart count() by <field> order by count() desc limit 10**.

Execute a query and click the search result visualization to dive deeper. In the context menu, click the field under the Top 10 Drilldown by section. The search result of the drill-down appears on the screen.

New query:

```
action = denied| chart count() by action order by count() desc
limit 10
```

3.8 Drilldown from Log Results

You can refine your search query by clicking the content of the results (key-value pairs or raw log messages) after the search has been done. Clicking on any value in the result adds a filter component to original search query. You can combine any number of filters, thereby making complex drill-down actions. The filter components (key-value pairs or raw log messages) are highlighted in the results as you drill down deeper. If you want to undo the drill-down on any component, click it.

For example, if you want to view successful login events for the user **rst@SLS.com** from the IP: **192.168.2.20**, click **successful login** in the action field and click the user **rst@SLS.com**. Finally, click IP: **192.168.2.20**. The clicked value is added to the query and is displayed in the query bar.

The example above is for the drill-down search conducted on the *filesystem*, the *SLS*, and the *localhost* respectively. Note that the filter components **"device_name"="localhost"**, **"collected_at"="SLS"**, and **"col_type"="filesystem"** automatically appear in the search query.

You can also carry out a negative drill-down search in the same manner. However, in this case, you have to use the **Shift** key while selecting the filter components (key-value pair or raw log messages).

3.8.1 Actions in the Field-Value Pairs

Once you execute a search query, you can apply various actions to the key-value pairs displayed. Click the drop-down menu on the key-value pairs to view the actions.

Top 10 Fields

You can view the **Top 10 Fields** for the selected fields and values. If you want to view the results for the particular search field, click *for this search*, else, click *for the whole database*.

Time Trend for Fields

You can view the **Time Trend** for the selected fields. If you want to view the results for the particular search field, click *for this search*, else, click *for the whole database*.

Time Trend for Full Resultset

You can view the **Time Trend** for the full result-set. If you want to view the results for the particular result-set, click *for this search*, else, click *for the whole database*.

Exclude field

You can perform a negative drilldown of the specific field-values by clicking the **Exclude fields** link. For example, if you click **Exclude** from the drop-down menu of the result set, **Value=read2**, all the results containing the field-value "read2" are removed.

The example above describes the **Exclude** operation on the value **read2**.

> **NOTE**
>
> The query **-("Value"="read2")** is automatically appended in the search query after clicking **Exclude**.

Explore in Search Template

You can drill-down any value in the search results directly into a search template. Clicking the **Explore in Search Template** option redirects you to the search template with the selected value filled in the corresponding field.

The **Explore in Search Template** option appears only for the search templates that contain the selected field in their respective **Fields** section.

Request for field

This option is applicable for the key-value pairs which are included under the Data Privacy Module. Clicking this option opens the **Data Privacy Request** panel from which you can make a request to view the decrypted values of the encrypted fields. After a request is accepted by a granting user, you can search for the specific field.

To view the decrypted key-value pairs, follow the steps given below:

1. Go to `Settings >> Configurations >> Data Privacy Module`.
2. Click the *Search* icon for the granted field under the **My Request** tab.
3. SLS redirects you to the **Data Privacy Search** from where you can view the decrypted values.

Add this field to interesting fields

You can select **Add this field to interesting fields** from the drop-down menu on the key-value pairs to add the required field in the *Interesting Fields* window. The fields added from here can be seen in the **Add Fields** panel of the **Interesting Fields** window.

Hide Fields

You can select **Hide this field** from the drop-down menu on the key-value pairs to hide the required field value(s). You can also hide the fields by going through the `My Preferences >> Search >> Search Log Fields` and entering the field name(s) in **Hide these Fields** text box.

Recover Hidden Fields

- Click the *User* drop-down menu at the top-right corner of the interface and select **My Preferences**.
- Select **Search**.
- Under **Search Log Fields**, deselect the hidden fields from the **Hide these Fields** text-box to unhide the fields.

Display maximum

Select a value from the drop-down menu to view the specified number of logs per page. The default value is 25.

3.9 Search Environment

The Search landing page lets you execute different types of search. Click the **Search** on the navigation menu to view the search environment.

3.9.1 Search Bar

The **Search Bar** contains the following components:

- **Query Bar**: To enter the search query.
- **Repo Selector**: To specify the repositories in which to search the logs.
- **Time-range Picker**: To specify the time-range in which to search for the logs.
- **Search Button**: Click Search to search logs. Alternatively, you can press the **Enter** key to conduct the search.

3.9.2 Search Page

The Search landing page contains six sections. On each section, use the `filter` text-box to search through the respective components. Provide at least one matching search term in the text-box to search the query.

My Search History

Lists the recent search queries executed in the SLS. Click a **Search** to automatically feed it to the search bar and display the results accordingly.

My Saved Searches

Lists the saved search queries in the SLS. Click a **Saved Search** to automatically feed it to the search bar and display the results accordingly.

Search Templates

Lists the search templates created in the SLS. Clicking a template opens the search template page. Enter the desired values in the *Update Parameters* section, select the required **Repos** and click *Update* to refresh the previously created widgets.

Labels

Lists the labels of the system. Labels are assigned while writing signatures for the logs. Click a label to automatically feed it in the search bar and display the results accordingly.

Vendor Searches

Lists the search queries provided by Stormshield. Click a search query to automatically feed in the search bar and display the results accordingly.

Search Views

Lists a maximum of 20 recently created views. Click any of the added *Search Views* from the list, or, *All search views* to view the search results.

3.10 Best Practice

- Use appropriate filters when using the chart and timechart commands for quick visualization.
  Example: **severity<3 device_ip IN BLACKLIST_IP | chart count() by severity, device_ip**
- Use the **rex** and the **norm** commands when you want to assign a value of a raw log message to a user-defined field. The **rex** and **norm** commands are like dynamic signatures written in a search.
- Select appropriate **Repos** and **Time range** to get quick search results.
- Avoid using commands such as **| search**, **| rename** that use the resources heavily.

4. Dashboard

A **Dashboard** is a data visualization monitor which updates regularly in real time. The drop-down menu beside this tab lists the Quick-links to various dashboards in the system.

The **Filter** option in the menu allows you to search for a desired dashboard.

You can add multiple widgets in a dashboard. A widget can hold charts, tables, and graphs generated by a search query. Each dashboard contains one or more panels, which contains charts, lists, and tables. If needed, you can change the height, width, and positioning of the widgets.

4.1 Creating a Dashboard

1. Go to `Dashboard`.
2. Click + to open the *Add Dashboard* panel.
3. Enter the **Dashboard Name**. You can also pull dashboards from the tabs on the left of the panel.
4. Click **Ok**.

4.2 Dashboard Tools

4.2.1 Add Widget

Widgets are panels that you can use to monitor the logs in real-time. You can personally set up a widget and add it to a *Dashboard* of your choice. For example, if you want to monitor the firewall activities of devices, create a widget with the search queries related to the firewall.

1. Go to `Dashboard`.
2. Select a **Dashboard** of your choice.
3. Click **Add Widget**.
4. Enter a **Name** for the widget.
5. Enter a **Query**. Alternatively, click **Select** to choose any query from the **Advanced Query Picker**.
   If you choose the *Advanced Query Picker*, select a query from the lists provided.
6. Choose the **Repos** from where you want to generate the logs.
7. Select a **Limit** for the number of logs to be generated.
9. Enable *Expose widget to public URL?* to share the widget publicly.

> **NOTE**
> - When exposed to public, the widget has an additional option titled *Open public URL*. Clicking this opens the search result in a new window.
> - The user the widget is shared with does not need the credentials to view the shared widget.

10. Provide a **Description** for the widget.
11. Select a **Time Range** for the logs in the repos.

NOTE

- You can set a time range of either only minutes or only day and hour.
- The maximum limit of the time range for the **day** field is 30.

12. Click **Finish**.

NOTE

In the Data Privacy Module enabled systems, the users who **Can Request Access** can view the values only in the encrypted form. The values cannot be decrypted to their original form.

IMPORTANT

While configuring repos for a new dashboard, make sure you select just the concerned repos, else, it might impede the system performance.

4.2.2 Report

You can generate reports from a particular dashboard. These reports are the replica of the positioning and placement of activities as set in the widget(s). From here, you can neither schedule nor change the layout of the report.

To create a report using a Dashboard, refer to Creating a Report from Dashboards.

4.2.3 Change Repos

Using this option, you can change the **Repos** for the log results of all the *Widgets* in the Dashboard.

4.2.4 Auto Arrange

You can manually change the sizes and positions of all the widgets in the dashboard. However, if you click **Auto Arrange**, SLS automatically arranges the widgets efficiently.

4.3 Editing Widgets

To **Edit** a widget, hover at the top-right corner and click the **Widget Options** icon. You can view the following editing options for the widget. Use these options to make the changes.

You can **Search** for the results, get **Info**, **Edit**, **Remove**, and **Enable/Disable** the widget.

Widget component lets you create graphs. You can monitor various activities through Table, Area chart, Line chart, Bar chart, Column chart, Gauge chart, Display chart, and Donut chart. However, the display types that are available for the results depend on the search you are making.

4.3.1 Tables

The following query gives the output shown above.

```
| chart count() by action
```

4.3.2 Area Chart

The Area chart is used to represent quantitative data graphically. The graph is used to interpret the quantitative statistics graphically. The graph is based on a Line graph, and the area between the x-axis and lines are emphasized with colors, textures or hatchings.

Area charts are used to represent accumulated totals using numbers and percentages. It is also used to show the trends over time along with all related attributes.

The x-axis of the Area chart represents the grouping parameter(s), and the y-axis represents values of the aggregation parameter.

The following query gives the output shown above.

```
| timechart count()
```

4.3.3 Line Chart

The Line chart displays information as a series of data points called markers. The markers are connected to each other by a line.

The Line chart consists of two axes, in which x-axis contains the value of the grouping parameter(s) and the y-axis contains the values of the aggregation parameter. It is similar to a Column chart, except that, a Column chart usually displays discrete values, whereas a line chart visualizes a trend in continuous data.

The following query gives the output shown above.

```
source_address=* | chart sum(datasize) as Datasize by source_
address
```

4.3.4 Column Chart

The Column Chart is a vertical bar graph that represents categorical data in rectangular bars with heights proportional to the values that they represent.

The Column Chart shows comparisons among discrete categories. It is a two-dimensional graph in which one axis of the graph shows the specific groups being compared and another one represents the measured value.

The following query gives the output shown above.

```
| chart count() by action limit 5
```

4.3.5 Bar Chart

The Bar chart is a horizontal bar graph that visualizes categorical data in a rectangular bar with the width proportional to the value.

In a Bar Chart, the x-axis represents the aggregation parameter and the y-axis represents the grouping parameter(s). Besides this, it is similar to the Column Chart.

The following query gives the output shown above.

```
| chart count() by col_ts  limit 5
```

4.3.6 Heatmap Chart

Heatmaps visualize individual values contained in a matrix and represent them through different colors. You can use it to reveal patterns, analyze similarities between variables, and to detect correlations.

The following query gives the output shown above.

```
| chart count() by action, protocol
```

4.3.7 Gauge Chart

Gauge chart presents the actual data in number and percentage.

The following query gives the output shown above.

```
| chart count()
```

Click the **Settings** icon at the top-right corner to change the *Rendering Parameters*.

For better visibility of the result, configure the starting values of the yellow and red colors. The final output of the gauge chart is a gradient that varies from green to red.

The data collected is shown in number and percentage. The number is converted into a percentage by the given **Max Value**. The default percentage of **Max Value** from where the **Red Starts** and **Yellow Starts** are 90% and 70% respectively.

4.3.8 Display Chart

The following query gives the output shown above.

```
| chart count(), max(datasize), avg(datasize)
```

Click the **Settings** icon at the top-right corner to change the *Rendering Parameters*.

Customize the output of the display by configuring the **Output format**. In this section, you can choose the fonts and color of the result. Additionally, you can use the display template similar to “jinja” to customize the search result.

> **NOTE**
>
> The **Output Format** section is disabled when you select the default layout.

4.3.9 Donut Chart

The following query gives the output shown above.

```
source_address=* | chart sum(datasize) as Datasize by source_
address
```

4.3.10 Clustered Column Chart

The following query gives the output shown above.

```
| timechart count(), avg(datasize)
```

You can use this chart to display the following response types:

1. Multiple Aggregation without Grouping
2. Multiple Aggregation with Grouping
3. Timechart Multiple Aggregation without Grouping
4. Timechart Multiple Aggregation with Grouping

4.3.11 Clustered Bar Chart

The following query gives the output shown above.

```
| chart count(), avg(sig_id) by action
```

You can use this chart to display the following response types:

1. Multiple Aggregation without Grouping
2. Multiple Aggregation with Grouping

4.3.12 Clustered Line Chart

The following query gives the output shown above.

```
| chart count(), avg(sig_id) by action
```

You can use this chart to display the following response types:

1. Multiple Aggregation with Grouping
2. Timechart Single Aggregation with Grouping
3. Timechart Multiple Aggregation without Grouping

4.3.13 Stacked Area Chart

The following query gives the output shown above.

```
| chart count(), avg(sig_id) by action
```

You can use this chart to display the following response types:

1. Multiple Aggregation with Grouping
2. Timechart Multiple Aggregation without Grouping

4.3.14 Stacked Column Chart

The following query gives the output shown above.

```
| timechart count() by action
```

You can use this chart to display the following response type:

1. Timechart Single Aggregation with Grouping

4.3.15 Radar chart

The Radar chart is a graphical representation of multi-variate data in the form of a two-dimensional graph, in which one or more quantitative variables are represented on axes starting from the same point.

The Radar chart is best for visualizing outliers in a dataset, especially for operation-related analysis such as performance metrics and quality improvement.

The line between the origin points and radii can be used as the scale for data points.

The following query gives the output shown above.

```
| chart count by action()
```

You can use this chart to display the following response types:

1. Single Aggregation with Grouping
2. Multiple Aggregation with Grouping
3. Timechart Single Aggregation without Grouping
4. Timechart Multiple Aggregation without Grouping

4.3.16 Parallel Coordinate chart

The Parallel Coordinate graph is a visualization technique used to plot individual data elements across multiple dimensions. The charts are ideal for comparing many grouping parameters and analyzing the relationships between them. Each grouping parameter has its axis, and all the axes are placed in parallel to each other. Values are plotted as a series of lines that are connected across all the axes. This means that each line is a collection of points placed on each axis, which have all been linked together.

The Parallel Coordinate chart shows both the forest and the tree. You can see the big picture in the patterns of the lines. You can highlight the individual lines to see the performance of a specific value of parameters. It is useful in the situations when the behavior of particular parameters may not be of concern, but a combination of those parameters may emphasize an abnormal pattern or relationship.

The following query gives the output shown above.

```
| process geoip(source_address) as source_country | chart count()
by source_country, sub_category, destination_
location
```

NOTE

1. Each line represents a relationship between two parameters rather than a trend or change in value.
2. As the number of values increase, the graph may be cluttered or may even overlap at times, which makes it difficult to perceive. In such a case, use the Brushing feature to highlight an individual or a group of values for better understanding.

3. You can view the value of the aggregation parameter by hovering over a relationship line.

Operations

Brushing

The Brushing feature eliminates one of the primary drawbacks of the Parallel Coordinate chart. When the number of data items in a Parallel Coordinate chart gets very high, lines get cluttered and overlap with each other. It makes the chart difficult to understand. Using the brushing feature, select an area containing one or many data points.

The lines under the brushed area are highlighted. You can then view the details of the relationship by hovering over the particular line. Additionally, you can drill down to the aspects of the relationship by clicking it.

Combined Drill-down

In addition to the regular drill-down operation, you can also perform a combined drill-down using the **Brush**. SLS performs the drill-down operation when you select a range of values in multiple axes using a brush and click the brushed area.

The results of the drill-down filters down to the combination of the selected grouping parameter values.

Changing order of the parameter

By default, the first grouping parameter of the query is assigned to the first axis of the Parallel Coordinate chart, followed by the other grouping parameters. You can change the order of a parameter by dragging it across the parameter with which you want the value to be exchanged.

Use The Parallel Coordinate chart to display the queries belonging to **Single Aggregation with Grouping** response type.

4.3.17 World Map

A World Map is a map of a country, a continent, or a region, with colors and values assigned to specific regions. Values are displayed as a color scale, and you can see the name of the country by hovering over a particular part.

In Single Aggregation with Grouping, the color shade on each region of a World Map displays the value of the aggregation parameter, i.e., higher the value of the aggregation parameter, darker the color.

In Multiple Aggregation with Grouping, the color shade on each region of a World Map displays the value of the first aggregation parameter. The values of all other successive aggregation parameters can be viewed using the sub-charts.

The following query gives the output shown above.

```
| process geoip(destination_address) as country_name | chart
count(), avg(datasize) by country_name, action
```

NOTE

1. Sections of a graph are clickable only if they have some value of aggregation parameter and the search query contains two or three grouping parameters. This means that you cannot click and drill down on the chart for a query with either a single grouping parameter or more than three grouping parameters.
2. For search queries with a single aggregation parameter and two grouping parameters, you can see a Donut chart by clicking on any region of a World Map (with some value for the aggregation parameter).
3. For search queries with a single aggregation parameter and three grouping parameters, you can see a Heatmap by clicking on any region of a World Map (with some aggregation parameter).
4. For search queries with multiple aggregation parameters and two or three grouping parameters, you can see a Clustered Column chart by clicking on any region of a World Map (with some value for the aggregation parameter).
5. You can drill down further from these sub-charts.

Operations

Rendering Parameters

Click the **Settings** icon at the top-right corner to change the *Rendering Parameters*.

The rendering parameters provide custom settings options to view data in different formats.

Choose a **Country** to specify the grouping parameter. Select colors in the **Positive Value** and **Negative Value** boxes to represent the positive and negative values of the aggregation parameters respectively.

Pan and Zoom

The **Pan and Zoom** feature lets you zoom in and out on a specific section on the world map and shift from one section to another.

NOTE

For more details on Operation of Worldmap Chart, refer to the **Search** section in the **User Manual**.

You can use this chart to display the following response types:

1. Single Aggregation with Grouping
2. Multiple Aggregation with Grouping

4.3.18 TreeMap

The **TreeMap** chart visualizes the hierarchical structure of a tree diagram. It displays the weight of each node in the form of the area size. Each node is assigned a rectangular area with their child nodes nested inside. The space of each node inside a parent node is displayed with proportion to all other nodes within the same parent node. If the weight of a child node is zero, the node is not included in the diagram.

The following query gives the output shown above.

```
source_address=* action=* | chart count() by source_address,
action order by count() desc limit 10
```

The first grouping parameter is the parent node of a TreeMap diagram, and all its successive parameters are the children nodes.

The name of the first grouping parameter is displayed in the breadcrumb whereas all its fields are displayed in the containers as individual nodes.

You can use this chart to display the following response types:

- Single Aggregation with Grouping

NOTE

The aggregation parameter determines the area size of each node in the container.

Operations

Zoom In and Zoom Out

The expanded diagram displays the node of the successive grouping parameter associated with the selected parent node. The new node is shifted to the breadcrumb, and the container is updated with the fields of the node in the breadcrumb.

Rendering Parameters

Click the **Settings** icon at the top-right corner to change the *Rendering Parameters*.

Choose one of the following:

- Single: All the nodes in the container are represented by a single color. You can select the color to represent the nodes from the **Color** picker tool.
- Unique: All the nodes in the container are represented by a unique color. In this case, SLS selects the colors randomly.
- Gradient: The **Color High** represents the node with the most significant area size and the **Color Low** represents the node with the least area size.
  Each section has a defined color, and different shades of the color represent all the nodes of the division. The darkest shade represents the node with the most significant area size, and the shade of the color fades as the area size of the nodes decrease.

NOTE

For more details on Operation of Treemap Chart, refer to the **Search** section in the **User Manual**.

4.3.19 Sankey chart

Sankey chart is a flow diagram used to depict a flow from one set of values to another. The connected values are called *nodes* and the connections are called *links*. It displays the corresponding grouping parameters on top of each node of the chart. The width of the link shows the magnitude of the flow. Colors are used to divide the diagram into different nodes or to show the transition from one state of the process to another.

Use the Sankey chart to show a *many to many* mapping between two or more nodes. The *aggregation parameter* is used to define the width of the flow between a source node and the destination node.

Example:

```
| process geoip(source_address) as country | chart count() by
country, severity, category, sub_category
```

You can use this chart to display the following response type:

- Single Aggregation with Grouping

Operations

Vertical Reposition

You can change the vertical position of the nodes by dragging them in the upward or the downward direction. You can either overlap the nodes or place them distinctly.

4.3.20 Day/Hour Heatmap

Heatmaps are used to visualize individual values contained in a matrix and represent them using different shades of a single color.

The Day/Hour Heatmap is an extension of a regular heatmap in which results are displayed in the day/hour format. It has seven rows and 24 columns. Each row represents a day of the week and each column represents an hour of the day. Therefore, each cell represents a single hour of a particular day.

The Day/Hour Heatmap only works for the Timechart Single Aggregation with Grouping response type with **every 1 hour** suffixed to the query.

Values of the aggregation parameter are displayed in the cells as per their timestamps.

The intensity of the color depends on the relative value of the aggregation parameters.

Example:

```
| timechart sum(datasize) as TotalDatasize every 1 hour
```

If the selected time-range is more than a week, a slider appears at the right end of the container. It allows you to slide over the days.

Operations

Rendering Parameters

You can assign custom colors to the Day/Hour heatmap for both positive and negative values. SLS uses the selected color to represent the value of the data obtained. The transparency of the color increases with a decrease in the value.

4.3.21 Bubble chart

The Bubble Chart is used to compare the relationship between variables in three dimensions: the x-axis, the y-axis, and the bubble radius. In the chart, the x-axis and y-axis represent the data location while the bubble radius compares the data size.

The following query gives the output shown above.

```
timechart count(), avg(sig_id) by action
```

You can use this chart to display the *Timechart Multiple Aggregation with Grouping* and *Multiple Aggregation wit Grouping* response types.

Operations

Rendering Parameters

The Bubble Chart accepts multiple parameters but uses only three parameters to form the chart. While the x-axis represents the timestamps by default, you can choose the parameters for the y-axis and the bubble radius. Hover over the chart legend and click on the Settings icon. Select the parameters for the y-axis and the bubble radius from their respective drop-down menus.

4.3.22 ATT&CK chart

The **ATT&CK** chart is a heatmap describing the attacks carried out on a system in the form of the attack tactics and techniques described by MITRE . The chart is displayed if the grouping parameter contains the **attack_id** field in the **Single Aggregation with Grouping** response type.

4.4 Managing Dashboards

4.4.1 Types of Dashboards

There are four types of dashboards in SLS.

- My Dashboards: The dashboards you created. From the *Actions* column, you can **Clone**, **Share/Unshare**, **Lock/Unlock**, and **Delete** the dashboards.
- Used Dashboards: The dashboards you used.
- Shared Dashboards: The dashboards shared by other users in the system. Click the **Use** icon in the *Actions* column to use the dashboards.
- Vendor Dashboards: The dashboards provided to you by SLS. To use these dashboards in your system, click the **Use** icon in the *Actions* panel. To clone the dashboards, click the **Clone** icon.

4.4.2 Exporting Dashboards

1. Go to `Settings >> Knowledge Base >> Dashboards`.
2. Select the dashboards that you want to export.
3. Click **Export**.
4. Save the **pak** file as a backup or store in the computer system to use it in another SLS.

4.4.3 Importing a Dashboard

1. Go to `Settings >> Knowledge Base >> Dashboards`.
2. Click **Import**.
3. Browse and upload the **pak** file containing the dashboards to import.
4. Click **Upload**.

NOTE

- Once you use a shared dashboard, its copy appears in the **My Dashboards** page. You can edit and share this dashboard with other users. If you need the initially shared dashboard version, delete your copy of it and make a new copy of the same dashboard from the **Shared Dashboard** tab. The same applies to the dashboards from the vendor section.
- You can drag and drop the widgets from one dashboard to another. However, you must avoid dropping the widgets into a locked dashboard.

4.5 Customizable Drilldown from Dashboard Widgets

Customizable drill-down options are available in dashboard widgets. It is possible to dive deeper by clicking the presented search results. The options to dive into specific parts of the result depends on the type of the search query. For example, while viewing a search result that includes fields such as *destination_address*, *destination_port*, *source_address*, and *source_port* in the query, it is possible to drill down the results based on these parameters.

The process of drill-down from a dashboard widget is the same as that of the search query. Two typical scenarios that cover all the aspects of search visualization from dashboard widget are described in this section.

4.5.1 Non-Empty Search from Widget

Consider a widget with the following search query.

```
destination_address=* | timechart count() by destination_port
```

The query displays the following visualization in the widget.

You can toggle between the *edit* and *non-edit* mode using the **widget options**. From the edit mode, you can choose from the available representations for the search results (in this case Clustered Line, Stacked Column, and Table). In the non-edit mode, you can carry out the drill-down process.

Select the section of *destination_port: 80* and *count(): 3* for the drill-down purpose.

When you click on the highlighted section of the result, a context menu appears with the options to further drill down on the specific parameters which are:

1. Filter
2. Drill down by
3. Top 10 drill-down by

The **Filter** type drill-down searches on *Range*, *destination_port* and *count()*. The **Drilldown by** and **Top 10 drill-down by** types drill-down searches on the *destination_address*.

The results of all three types of drill down can be opened and viewed in the same window or a new window. The drill down on *Range* value (either on full or partial time-range) can be enabled or disabled by clicking the toggle icon.

While performing the drill-down on the *Range* value, the search results for the time-range opens on the same page. The search results for the *enabled* and *disabled* time-range are shown below:

Likewise, when the drill-down is carried out on the *"destination_port"=138*, the search result for the destination port opens in the same page.

When the drill-down is carried out on *"count()"=3*, the search results for the count of logs opens on the same page.

When the drill-down is carried out on the *destination_address*, the search result for the destination address opens in the same page.

When the Top 10 drill-down is carried out on the *destination_address*, the search result for the destination address opens in the same page.

4.5.2 Empty Search from Widget

Consider a widget that has no search query.

The blank query displays the following visualization in the widget.

The search result for a blank query is the logs collected for the specified range of time without any visualization. However, you can refine the search query by clicking the components of the search results such as a key-value pair, or a raw log message. Clicking on any value in the result opens a regular search with the selected parameter as the search query.

For example, if you click *syslog* on the search results:

This opens the search result of the query **"col_type"="syslog."** The search visualization depends on the chosen value.

From this point, a regular drill-down can be carried out. If you hover over any section of the search results, the related information of the section is summarized in a tooltip.

Clicking on the highlighted section opens a new dialog box with the same three drill-down options of Filter, Drilldown by, and Top 10 drill down by which can be carried out in the same way as described in the scenario-1.

5. Incident

Incidents are used to identify, analyze, correct, and thereby prevent information hazards in the future. SLS lets you find events such as a system crash, power down, cables unplugged, high disk usage, high CPU usage, and forensics by creating incidents for each of them.

Incidents can be created either on an ad-hoc basis from the search logs or by pre-defined alert rules.

If you create an alert rule to detect system crashes, an alert is fired whenever the search results match the alerting criteria. SLS then creates the corresponding incident based on the alert rule.

You can view the log source of an incident to determine if it was triggered by an alert rule or by a search query.

5.1 Creating an Incident

The methods of creating an incident are as follows:

- From Search Interface
- From Alert Rules

5.1.1 Creating Incident from Search Interface

You can create incidents for a particular search query from the Search Interface. Follow the instructions below to create incidents in this way.

1. Go to `Search`.
2. Execute a query to create its incident.
3. Go to the **Add Search To** drop-down menu and select **Incident** to open the *Create Search Incident Panel*.
4. Enter the **Incident Name**, and the **Description**.
5. Select a **Risk** level for the incident.
6. Select a user from the **Assigned to** drop-down menu to assign the ownership of the incident. The **Assigned to** drop-down menu displays all the distinct Users mapped to the Incident User Groups (via User Groups).
7. Choose a group(s) from the **Manageable by** tree node structure. The tree node structure displays all the Incident User Groups with their corresponding users present in the system. Users selected in both the **Assigned to** and **Manageable by** sections can view the generated incident, reassign it, and comment on the data. However, only the **Assigned to** user can resolve it.

> **NOTE**
>
> - While creating the incident, you can only see the **Assigned to** and the **Manageable by** sections if you belong to any of the Incident User Groups.
> In this case, you are assigned to the generated incident, and you are responsible for managing it.
> - If required, you can assign an incident to yourself and select none of the Incident User Groups from the **Manageable by** tree node structure.

8. Click **Submit**. As soon as this form is successfully submitted, a new incident is generated and populated in the Incident feed.

> **NOTE**
>
> - The **Assigned to** and the **Manageable by** sections appear the same to the LDAP Users.
> - The Alert Rule/Incident creators can see the incidents generated even if they are not present in the **Assigned to** drop-down menu and the **Manageable by** tree node structure.

5.1.2 Creating Incident from Alert Rule

The purpose of an alert rule is to monitor data continuously. Once SLS finds the search result matching an alert, it fires the corresponding incidents. The process of creating an incident from alert rules is given below:

1. Go to `Settings >> Knowledge Base >> Alert Rules`
2. Create an alert rule on the basis of your requirements.
   For details, refer to the **Creating an Alert Rule** section under Administration Manual.
3. After creating the alert, click the bell shaped **Setup Notification** icon in the *Actions* column.
4. Choose the type of notification you would like to configure and fill in their respective required parameters. Refer to the *"Setting Up Alert Notifications"* section under Administration Manual for the detailed information.
5. Click **Save**.

After creating the alert rule, the incidents of the corresponding alerts fired are automatically generated and populated in the Incident menu.

5.2 Filtering an Incident

There may be numerous incidents triggered by the devices which in turn, can search for a particular incident a complex task. To narrow down the search for a particular incident, you can use various filters such as **Name**, **TimeRange**, **Users**, **Risk**, **Type**, and **Status**.

The search for a particular incident becomes easier if you specify its name. You can also use the following filters to search for the specified incidents.

- **TimeRange**: to view the incidents generated at a particular time.
- **Users**: to view the incidents assigned to you or any other users.
    - When you select the **All Incidents** option, the incidents created by, assigned to and manageable by the current user (the user who has logged in) are listed.
- **Risk**: to view the incidents of a particular severity level (critical, high, medium, low).
- **Type**: to view the incidents by the source (alert, search) via which generated them.
- **Status**: to view the incidents according to their status (resolved, unresolved, closed).

5.3 Incident Actions

In the Incident feed, you can find the list of all the incidents along with their states. You can **Resolve**, **Re-open**, **Close**, **Comment** on, and **View the Data** for these incidents.

5.3.1 Resolve

Once appropriate action(s) has been taken on a particular incident, you can **Resolve** it.

5.3.2 Re-open

If you feel that an incident has not been satisfactorily resolved even after it was closed, you can re-open it. This can be done by clicking **Re-open** on the particular incident.

5.3.3 Close

After an incident is resolved and needs to be close, you can close it by clicking on the **Close** option. Once an incident is closed, it is not shown in the incident feed. However, it can easily be retrieved using the **Closed** option in the **Status** filter.

5.3.4 Comment

You can post comments on the incidents seen in the incident feed. You can also track the actions taken over the incidents via the comments.

5.3.5 View Data

The **View Data** option directs you to the search page and shows the log messages that triggered the incident.

5.3.6 Incident Data

The **Incident Data** option opens a pop-up panel to display the data of the incident in the format specified in the **Incident Data View** panel while creating the alert rule.

> **NOTE**
>
> If the format was not specified in the **Incident Data View** panel while creating the alert rule, the **Incident Data** panel displays the logs of the generated incident.

5.3.7 Assign to me

The **Assign to me** option assigns the incident to you (the user who is logged in).

5.3.8 More

The **More** drop-down menu near the top-right corner of the Incident page lists six more actions.

You can resolve some selected incidents or all the incidents at once using the **Resolve Selected** and **Resolve All** options respectively.

You can close some selected incidents or all the incidents using the **Close Selected** and **Close All Resolved** options respectively. Please note that the incidents **cannot** be closed without being resolved first.

Also, you can reassign some selected incidents or all incidents at once using the **Reassign Selected** and **Reassign All Selected** options respectively. Reassigning opens a window prompting you to select a user to reassign the incidents to.

6. Report

A SLS report is the collection of information, events, and findings which are collected, analyzed, and presented in an organized manner. You can view all the generated reports, rules to generate the reports, and the report templates in the **Reports** page.

The drop-down menu beside this tab lists the recently generated reports and their formats. It also contains the quick-links to **Add New Report**, **Report Inbox**, **Archived Reports**, **Approved Reports**, **Flagged Reports**, **Shared Reports**, **Report Jobs**, **Report Templates**, and **Layout Templates**.

The **Filter** option in the menu lets you search for a desired report.

In SLS, you can generate reports automatically as well as manually. You can specify the duration of the log activities for the report generation and also pinpoint its time as per your requirement. The generated reports are populated in the inbox along with their names and their corresponding formats (PDF, XML, HTML, DOCX, or CSV). Additionally, you can get a copy of the report in your e-mail upon specifying the report recipients. With the exception of ad-hoc reports, all the reports can be zipped and emailed.

NOTE

The generated reports only contain the first 1000 logs. If your report is likely to have more than 1000 logs, use the **Search Templates** option under `Settings >> Knowledge Base`.

6.1 Creating Reports

In SLS, reports can be created in 3 different ways.

- From a Search Query.
- From Dashboards.
- From a Report Template

6.1.1 Creating a Report from a Search Query

1. Go to `Search`.
2. Enter a **Search Query**.
3. Click the **More** drop-down menu.
4. Select **Report** to open the *Create Report* panel.
5. Provide a **Name** and an **E-mail Address**.
6. Click **Submit**.

6.1.2 Creating a Report from Dashboards

1. Go to `Dashboard`.
2. Click **Report** to open the *Create Report* panel.
3. Enter a **Name** and an **E-mail Address**.
4. Click **Submit**.

6.1.3 Creating a Report from a Report Template

By using a **Report Template**, you can specify the format and the design of the report. However, to generate a report using this method, you need to first define parameters such as panels and headers of the template itself.

Query Selection

To successfully generate a report, you need to first provide a correct and valid query. You can either create a new query on your own or choose one from the history.

1. Go to `Report`.
2. Click **Report Templates** under the *Reporting* menu on the left.
3. Click **Add**.
4. Enter a name for the *Report Template* by clicking the **write** icon.
5. Enter a valid query in the **Add Query** tab or click **Select Query** to open the *Advanced Query Picker*. If you select the **Advanced Query Picker**, choose queries from the provided lists.

> **NOTE**
> - While selecting a **Live Search** from the **Advance Query Picker**, the title of the live search, created from the `Dashboard`, is automatically entered in the panel header field.
> - If you are using a **timechart** query with grouping parameters and it has more than ten values, the report is generated with an error message. This is done so as not to affect the rendering properties of the graph in the report.

6. Click **Add** to open the *Rendering Options* panel.
7. Provide a header on the **Panel Header** textbox.
8. Choose a **Limit** for the number of logs.

> **NOTE**
> **Limit** is visible only for non-aggregate commands.

9. Choose one of the **Component Options**.

> **NOTE**
> If you choose the **Chart** component option, choose the **Chart type** from the drop-down menu.

10. Click **Ok**.

Scheduling

Using this option, you can schedule the generation of reports.

1. In the *Scheduling* panel on the right, click **Add**.
2. Choose the options in the drop-down menu to schedule the report generation to the minute. If you choose the **Monthly** tab, SLS creates a link titled **Monthly (Last 30 days) on 1st day, 0th hour**. Click the link to open the *Monthly Scheduling* panel.
   - Select a **Time Range**.
   - Select a *Day* and an *Hour* in which to run the report generation process.

4. Select a **Time Zone**. The scheduling time information and the report results are generated in the selected timezone.
5. Select **Repos**.
6. Select the formats in which you want to generate the report.
7. Check the **Send email** option and enter the e-mail addresses and *Subject* if you want to send the report as an e-mail.
8. Select the **Send compressed report on email** option to get a compressed report on your email.

> **NOTE**
> If you choose the **HTML** format, the report is zipped regardless of the option you choose.

9. If the Data Privacy Module is enabled in the system, you can view the **Data privacy module** tab.
   - Check the **Generate report with original data** checkbox to generate decrypted values in the report.
   - Check the **Show raw logs** option to get access to the raw logs in the generated report.

> **NOTE**
> In the Data Privacy Module enabled systems, if you check **Generate report with the original data**, a request message is sent to the users with the **Can Grant Access** privilege. The request can be viewed under `Settings >> Configurations >> Data Privacy Module >> Pending Request`.
> The scheduled reports are generated only if the request is approved.

Choosing a Layout

1. Click the **Layout** option below the *Scheduling* tab.
2. Choose a **Layout Template**.
3. Enable the **Include Table of Contents** checkbox to add a table of contents in the report.
4. Select a **Component Placement** option,
   - If you choose **Use Default**, SLS positions the components in the report automatically.
   - If you choose **Personalize**, you can choose the placement of the components in the report.

Choosing the **Personalize** option opens the *Report Design* panel.

Report Design

The placement of the components depends on how you configure the panels and headers in the **Report Design** panel.

The panel is divided into two sections, **Structure** and **Layout Preview**. All the queries created earlier are listed under the structure section and the layout of your report can be previewed in the layout preview.

1. In the *Structure* section, click **Add**.
2. Choose the component that you want to add to the template.
   - If you choose a *Panel*, enter the **Position**, **Area 1 width**, and **Count**.
   - If you choose a *Header*, enter the **Position**, **Text**, **Font**, and **Color**.

   > **NOTE**
   > - **Area 1 width** does not appear for *Simple Panel*.
   > - You can also add a **Page Break** to the template. In this case, provide a **Position**.
3. The added components appear in the **Components** section.
4. To add components in a panel, click the ⊕ button for the respective panel. You can either add the results of a query or a paragraph.
   - To add a query, select a query listed under *Queries*.

     > **NOTE**
     > The *Queries* field is populated with only the queries that have not been added to the **Layout Preview**.
   - To add a paragraph, click the **Add** button. Enter the **Text**, **Font**, and **Color**.
5. You can add as many **Panels**, **Headers**, and **Paragraphs** that you want. However, you can add a single query only once.
6. **Edit** or **Delete** the components by clicking the respective buttons.
7. Click **Ok**.

6.2 Report Templates

6.2.1 Types of Report Templates

There are four types of report templates defined in SLS.

- My report templates: The templates you made.
- Used report templates: The templates you used previously.
- Shared report templates: The templates shared by other users.
- Vendor report templates: The templates provided to you by the vendor, i.e. SLS.

6.2.2 Running a Report Template

SLS automatically runs a *Report Template* based on the defined scheduling rules. You can also run it manually whenever you want. To run a template manually,

1. Go to `Report >> Report Templates`.
2. Click the **Run This Report** icon in the *Actions* column to open the *Run Report* panel.
3. Select the desired **Repos**, the **Time Zone**, and the **Time Range**.
4. Select one of the **Export Type** formats.
5. Enter the **Email** address(es), if you want to send the report as an attachment in the email.

6. If the Data Privacy Module is enabled in the system, you can to view the **Data privacy module** tab.
   - Check **Generate report with the original data** if you want to decrypt the fields in the report.
   - Check **Show raw logs** to access the raw logs in the report.
7. Click **Submit**.

NOTE

- In the Data Privacy Module enabled systems, if you check **Generate report with the original data**, a request message is sent to the users with the **Can Grant Access** privilege. The request can be viewed under `Settings >> Configurations >> Data Privacy Module >> Pending Request`.
  The report can be viewed or downloaded only after the request is approved.
- You can view ad-hoc or recurring reports by clicking the **Run** button inside the Template tab. Your settings for the scheduled reports are not overwritten.
- When you run a report manually, it does not tamper with the report's scheduled time in any way.

6.2.3 Sharing a Report Template

1. Go to `Report >> Report Templates`.
2. Click the **Share/Unshare** icon in the *Actions* column for the report.

NOTE

You can **Unshare** a report in the same way.

6.2.4 Exporting Report Templates

1. Go to `Report >> Report Templates`.
2. Select the report templates you want to export.
3. Click **Export**.
4. **Save** the exported package.

6.2.5 Importing Report Templates

1. Go to `Report >> Report Templates`.
2. Click **Import**.
3. Browse for the required report templates.
4. Click **Upload**.

6.2.6 Cloning a Report Template

1. Go to `Report >> Report Templates`.
2. Click the **Clone Report** icon under the *Actions* column for the template.
   - To clone multiple Report Templates, select the respective templates. Click the **More** drop-down menu and choose **Clone Selected Report Templates**.
   - To clone all the Report Templates, click the **More** drop-down menu and choose **Clone All Report Templates**.
3. Enter a new **Name** for the cloned template.
4. Select the **Replace Existing?** checkbox to replace an existing template with the same name.
5. Click **Clone**.

6.2.7 Deleting a Report Template

1. Go to `Report >> Report Templates`.
2. Click the **Delete** icon under the *Actions* column for the template.
   - To delete multiple Report Templates, select the respective templates. Click the **More** drop-down menu and choose **Delete Selected Report Templates**.
   - To delete all the Report Templates, click the **More** drop-down menu and choose **Delete All Report Templates**.
3. A delete confirmation dialog box appears on the screen. Click **Yes** to proceed.

6.3 Layout Templates

In SLS, a layout template is a structure based on which the appearance of a report is determined. You can create as many layout templates as you want and specify the layout for each report.

You can either import new layout templates into the system or create a personalized template.

6.3.1 Importing a layout template

1. Go to `Report >> Layout Templates`.
2. Click **Import** to open the *Import Layout Templates* panel.
3. Browse the desired layout template.
4. Click **Upload**.

6.3.2 Creating a layout template

1. Go to `Report >> Layout Templates`.
2. Click **Add**.
3. Provide a **Name** for the template.
4. Upload the **Background Image**, **Cover Page Image** and **Data Page Image** to be displayed in the report.

NOTE

- The **Name**, **Background Image**, **Cover Page Image**, and **Data Page Image** are mandatory fields.
- The maximum width and height allowed for each of the image are given below:
  - Background image = 595 * 842 pixel (width * height)
  - Cover Page Image = 300 * 128 pixel (width * height)
  - Data Page Image = 160 * 59 pixel (width * height)

5. Choose the **Footer Background Color**, **Footer Text Color**, and provide the **Footer Text**.
6. Click **Submit**.

6.4 Report Jobs

The **Report Jobs** section lists the reports currently being generated with their generation time, current status, and remarks.

NOTE

- Once a report is successfully generated, it is pushed into the inbox.
- This section also lists the reports that have failed to generate with the reasons for their failure. The reason for the failure is displayed under **Remarks**.

6.5 Generated Reports

This section is divided into **Inbox**, **Approved**, **Archived**, **Flagged**, and **Shared**.

Inbox lists all the successfully generated reports. You can either *approve*, *archive*, *flag*, *share*, or *delete* a report from the Inbox. The report is moved into the *Approved*, *Archived*, *Flagged*, or *Shared* pages or completely deleted from the system based on your action.

6.5.1 Archive

1. Go to `Report >> Inbox`.
2. Mark the reports that you want to archive.
3. Click the **Archive** button.

NOTE

- Important reports must be archived to save the inbox from getting crowded.
- You can see the archived reports in the **Archived** tab under *Generated Reports*.

6.5.2 Flag

1. Go to `Report >> Inbox`.
2. Mark the reports that you want to flag.
3. Click the **Flag** button.

> **NOTE**
>
> You can also flag a report by clicking the **Flag** icon in the *Actions* column.

6.5.3 Share

1. Go to `Report >> Inbox`.
2. Mark the reports that you want to share.
3. Click the **Share** button.

> **NOTE**
>
> - Other users in the SLS can view the shared reports.
> - You can also share a report by clicking the **Share** icon in the *Actions* column.

6.5.4 Approve

When a generated report is verified, you can push that particular report in the approved section for simplicity. It helps you organize the reports.

1. Go to `Report >> Inbox`.
2. Mark the reports that you want to approve.
3. Click the **Approve** button.

> **NOTE**
>
> You can also approve a report by clicking the **Approve** icon in the *Actions* column.

6.5.5 More

Under this dropdown you can **Mark as read**, **Mark as unread**, **Remove Flag**, **Unshare**, and **Disapprove** a report.

6.5.6 Delete

1. Go to `Report >> Inbox`.
2. Mark the reports that you want to delete.
3. Click the **Delete** button.

6.5.7 Activities

Using the **Activities** option in the *Actions* column, you can comment on a generated report.

1. Go to `Report >> Inbox`.
2. Click the **Activities** icon in the *Actions* column of the concerned report.
3. Enter a **Comment** and click **Submit**.

6.6 Cleanup Reports

Using the **Cleanup** option, you can either delete the reports generated before a certain date or delete reports within a specific date range.

1. Go to `Report >> Inbox`.
2. Click **Cleanup** to open the *Cleanup Old Reports* panel.
3. Select **Older than** to delete all the reports generated before the provided date.
4. Select **Date range** to delete all the reports within the provided date range.

NOTE

- The date range is inclusive.
- Go to **Cleanup Jobs** to see the status of all the initiated cleanups.

7. My Preferences

Go to **My Preferences** to customize your personal settings. For customization, click the drop-down menu with your username at the top-right corner of the interface and select **My Preferences**.

The different sections of the preferences pop-up panel allows you to customize your settings.

7.1 General

1. Select the *Show settings item help* to display help on items in **Settings** section. Hover over the items to see details of the intended Settings item on the right side of the screen.
2. Select a **Result limit** from the drop-down menu to specify the display limit of search results and the *Page size*. The *Page size* determines the maximum number of rows to be displayed per page in *Settings* and *Reports* pages.
3. Enable the **Pre compute dashboard data** option to be continuously prepare the result of the dashboard, even if the dashboard is not viewed.

7.2 Search

1. Enable **Display search help pop-up** to display the drop-down for search assistance while writing a query.
2. Enable **Hide Histogram in Search Page** as per your requirement.
3. Select **Disable Interesting Fields in Search Page** to hide the **Interesting Fields** window from the Search page.
4. Select the *Display All Available Fields* to display all the available log fields in the result of the query. Similarly, select *Display Minimum Fields* to display only the default fields.
5. Enter specific fields to hide in the **Hide these Fields** text box.

7.3 Change Password

Enter the **Old Password**, **New Password**, and **Retype** the new password to change your password.

7.4 Date Time

Select a **Time Zone**, a **Date Format**, and an **Hour Display Format**.

> **NOTE**
>
> The logs are collected in Coordinated Universal Time (UTC) irrespective of the **Time Zone** you select.

7.5 Notification

Select **Top Left**, **Bottom Left**, **Top Right**, or **Bottom Right** to position the notification.

7.6 Log Fetching Key

The *Secret Key* denotes a unique identification for a particular user. You can use this key to fetch logs from SLS using the APIs exposed by SLS.

8. Label Packages

Label Packages are collections of labeling rules. You can define the rules to automatically add labels to the incoming logs.

Consider the following log:

```
User Bob failed to login to the system, reason: account locked
out
```

You can define a labelling rule to assign the labels **Login_Fail** and **Locked** to this log. Use the search query **label = Login_Fail, Locked** to get all similar log results.

8.1 Applying Labels using Normalization Signatures

Once a label is applied to the log messages with signatures, it is associated with that particular signature.

1. Go to `Settings >> Knowledge Base >> Normalization Packages`.
2. Click the **Signatures** icon in the *Actions* column for the respective normalization package.
3. Click the **Edit Signature** icon in the *Actions* column for the respective signature.
4. Type **label** in the first textbox for **Key Value**.
5. Enter a list of labels in the second textbox.
6. Click **Submit**.

> **NOTE**
>
> You can also add labels while adding a normalization signature.

8.2 Applying Labels with Labeling Rules

1. Go to `Settings >> Knowledge Base >> Label Packages`.
2. Click the **Manage Labels** icon in the *Actions* panel for the respective label.
3. Click **Add** to open the *Search Label* panel.
4. Provide a suitable **Query**, a **Package Name**, and a **List of Labels**.
5. Click **Submit**.

In this example, all the log messages satisfying the search query **device_name = localhost** are labelled with **Localhost** and **127.0.0.1**.

8.3 Applying Labels from the Search Interface

1. Go to `Search`.
2. Enter the query to which you want to add the labels.
3. Click **Search**.
4. Click the **Add Search To** drop-down menu.
5. Select **Labelling Rule** to open the *Search Label* panel.

6. Select a **Package**, and enter a **List of labels**.
7. Click **Submit**.

> **NOTE**
>
> - Labels can contain only alphanumeric characters.
> - You can create and import labeling rules.

9. Appendix

9.1 Appendix: List of Fields

These field words are used to write signatures that normalizes the raw logs. Users are recommended to use these fields to write signatures.

A

```
access
access_control_list
access_list
access_mask
access_point
access_rights
account_name
acl_name
action
action_code
action_flags
action_group
action_id
actual_action
actual_mps
admin
admin_id
admin_name
agent
alarm_type
alert_id
alert_message
alert_name
alert_type
algorithm_name
another_patient
another_patient_id
answer
antivirus
application
application_action
application_category
application_hash
application_id
application_list
application_name
application_type
application_version
attack_group
attack_id
attack_message
attack_type
attacking_ip
attribute_name
audit_name
```

```
authentication_source
```

B

```
backend_name
branch
browser
bss_id
```

C

```
cache_info
caller_address
caller_computer
caller_database_user
caller_domain
caller_login
caller_logon_id
caller_object
caller_user
callout_name
cat_id
category
category_id
certificate_name
change_type
changes
channel
child_object_type
child_object_url
class_type
cleaned_items
cleanup_time
client
client_address
client_agent
client_domain
client_port
client_type
client_user
code
command
command_name
comment
company_name
component
component_version
computer_name
confidence
configuration_path
connection_id
connection_name
connection_time
consumer
consumer_id
content_type
```

```
context_name
correlation_id
count
counter
cpu_time
cpu_usage
current_time
```

D

```
database
database_id
database_name
database_option
database_principal_id
database_principal_name
datarate
datasize
datatypeid
date
default_context
deleted_datasize
deleted_mailboxes
description
destination
destination_address
destination_dns
destination_domain
destination_email
destination_interface
destination_ip
destination_location
destination_mac_address
destination_network
destination_object
destination_port
destination_release
destination_url
destination_zone
destinaton_address
detection_timestamp
device
device_address
device_address_ipv6
device_host
device_id
device_type
dhcid
direction
directory
disk
dispatch_type
display_name
disposition
DNS
doable_mps
```

```
document_id
domain
download_site
downloaded_by
dst_ip
dst_name
duration
duration_seconds
dvc_host
```

E

```
email_id
email_port
encryption
end_datasize
end_items
end_time
endpoint_address
endpoint_domain
error
error_code
error_message
error_messsage
establish_time
event
event_category
event_id
event_level
event_log
event_name
event_source
event_tag
event_time
event_type
exception
```

F

```
facility
failure_code
feature
file
file_name
file_path
file_size
file_type
files_skipped
filter
filter_action
filter_info
first_seen
flags
folder
folder_id
folder_name
```

```
free_size
from_release
frontend_name
function_name
funtion_name
```

G

```
gateway
generated_bytes
generated_records
generated_time
group
group_id
group_name
```

H

```
handle
handle_id
hardware_address
hash_key
hash_type
healthy
hierarchy
hip_name
hip_type
host
host_name
host_url
host_user
http
```

I

```
id
identity
imap
in_use
inbound_spi
infected
infection_destination
infection_destination_address
infection_source
infection_source_address
inserted_time
inspection_subrule
instance
institution
interface
interface_address
interface_name
interface_state
intrusion_id
intrusion_url
investigation_id
```

```
IoDepth
is_column_permission
```

J

```
job
job_id
job_name
```

K

```
kernel_time
key
```

L

```
last_update_time
layer_name
lease_address
lease_duration
license
life_id
link
local_address
local_proxy
location
log_ts
log_type
login_id
login_name
logon_id
logon_ID
logon_type
lower_limit
```

M

```
mac_address
machine
mail_id
mailbox
mailbox_guid
mailbox_owner
mailer
malware_id_mcafee
malware_id_sophos
malware_id_webroot
map
mapped_name
match_option
matched_criteria
mdb_guid
member_id
member_name
message
message_id
```

```
message_subject
message_type
method
mime_type
mobile_number
mobile_station
most_used
mount_point
msg_idmac_address
```

N

```
name
nas_address
nas_identifier
nas_port
nat_address
nat_destination_address
nat_destination_port
nat_source_address
nat_source_port
nc_address
net_mask
network_device_name
network_user
network_view
new_bandwidth
new_memory
new_policy
new_port
new_status
new_user
new_value
new_value_type
node
ns_name
number
```

O

```
object
object_count
object_id
object_name
object_server
object_source
object_type
object_url
occurrences
oid
old_bandwidth
old_memory
old_port
old_status
old_value
old_value_type
```

```
omitted
operation
operation_type
operations
outbound_datasize
outbound_spi
OutgoingConnections
owner
```

P

```
package
package_name
packet
pages_read
pages_updated
palyload_name
parent_network
parent_object_filepath
parent_object_subtype
parent_object_type
parent_object_url
parent_process
parent_process_id
passcode_type
path
patient_id
patient_name
payload_url
peer_address
peer_interface
peer_interface_state
peer_ip
peer_name
permission_bitmask
permission_level_id
physical_memory
physical_memory_percent
pin_code
policy
policy_applied
policy_id
policy_name
pool_name
pop3
port
port_channel
port_id
port_name
portnum
prevalence
previous_time
primary_dns_address
priority
private_address
privilege
```

```
probation_time
probe_name
process
process_id
process_name
processed_mailboxes
product
profile
profile_changed
profile_name
profile_used
property
protocol
protocol_id
protocol_map
provider_name
```

Q

```
query
query_result
queue_id
```

R

```
radioband
realm
reason
reason_code
received_datasize
received_packetsize
receiver
recipient
record
record_name
record_type
referer
relay_address
relay_interface
remaining_mailboxes
remote_address
remote_proxy
remote_user
repeat_count
repos
reputation
reputation_score
request_date
request_guid
request_id
request_method
requested_action
resolved_domain
resource
response_time
result
```

```
result_code
retained_datasize
retained_mailboxes
return_code
risk
risk_level
risk_name
risk_type
roaming
role
role_id
role_name
root_context
rows_deleted
rule
rule_id
rule_name
rulebase
```

S

```
scan_id
scanned
scanned_items
scanner_address
scanner_node
schema
schema_name
scheme
scope
search_id
search_name
secondary_actions
secondary_dns_address
secondary_user
security
sender
sensitivity
sensor_id
sent_datasize
sent_packetsize
seperated_mailboxes
sequence_number
serial
serial_number
server
server_address
server_host
server_instance_name
server_name
server_principal_id
server_principal_name
server_principal_sid
server_time
service
service_name
```

```
Service_start_type
Service_type
service_type
session
session_id
session_name
session_server_principal_name
sever_address
severity
shutdown_type
signature_id
site
size
size_limit
skipped_mailboxes
smoke
smtp
snapshot
snort_id
source
source_addresss
source_computer
source_destination
source_dns
source_domain
source_handle_id
source_host
source_interface
source_ip
source_location
source_mac_address
source_network
source_object
source_port
source_zone
spi
spi_code
spyware
ssid
start_datasize
start_items
start_time
state
statement
station_identifier
status
status_code
status_msg
sub_category
sub_status_code
subject
substatus_code
succeeded
synchronized_files
system
system_id
```

T

```
table
target_account
target_database_principal_id
target_database_principal_name
target_database_user
target_domain
target_entity
target_handle_id
target_login
target_logon_id
target_name
target_object
target_process
target_server_principal_id
target_server_principal_name
target_server_principal_sid
target_type
target_user
task
tenant
tenant_id
terminal
terrmination_state
thread_id
threat
threat_category
threat_id
threat_level
threat_severity
tid
time
time_zone
timerange_end
timerange_start
token_type
total
transaction_id
transport
type
```

U

```
unit
update
update_ts
upper_limit
url
url_category
url_category_name
url_query
url_tracking
use
```

```
used
user
user_time
user_type
```

V

```
value
vendor
vendor_id
virtual_address
virtual_firewall
virtual_memory
virus
virus_name
vpn_address
vpn_variant
```

W

```
waiting_time
web_domain
workstation
```

Z

```
zone
```

9.2 Appendix: Connections required by SLS

In some cases, you might have to configure your firewall for smooth operation of the SLS services. Some key points for firewall rules are listed below. Make sure to add the following rules in your firewall.

- Configure a firewall rule allowing port 1194/UDP to allow Open VPN required for accessing client SLSs.
- Configure a firewall rule allowing port 22/TCP to allow SSH connection.
- Configure a firewall rule allowing port 80/TCP and 443/TCP to allow HTTP connection.
- Configure a firewall rule to allow the SLS to connect to reverse.immune.dk at port 1193.

Apart from the ports mentioned above, some other ports might open as requested by different services.

9.3 Appendix: Additional Notes on SLS Query Language

Process or Count

Since **count** and **process** are keywords, they must be enclosed within double quotes.

```
MsWinEventLog product=* | chart count() as "Count" by product
order by count() desc limit 10
```

Similarly,

```
MsWinEventLog product=* "process"=* action=*
| fields product, "process", action, object
```

Conditional Expression

Conditional expression within parenthesis () must be separated explicitly by **or**.

```
| chart count(label = delete or label = remove) as remove
```

Forward Slash Expression

Any expression after the forward slash must also be enclosed within double quotes.

```
source_name = "/opt/immune/var/log/audit/webserver.log"
| chart count() by source_address
```

norm

```
| norm doable_mps=<dmps:'['0-9']'+>
```

```
| norm <:'\['><my_field:word><:'\]'> | chart count() by my_field
```

timechart

Limit does not work with timechart.

```
| timechart count() by col_type
```

Capturing normalized field values

You can use *norm on* command to capture the normalized field value in the log search result.

```
Suppose the log search result consists of a log value pair

source_name = /opt/immune/var/log/benchmarker

Now, if you want to capture the first two words of the path,
you can write the query as follows:

| norm on source_name <capture:'\/opt\/immune'>
```

This feature works well with rex command too.

```
user=* | rex on user:\s+(?P<account>\S+)@(?P<domain>\S+)
| chart count() by account, domain | search account=*
```

In the example above, the rex command is used on a field which captures email addresses. The email address is then broken into *account* and *domain* using the corresponding regex.

9.4 Appendix: Grok Patterns

The SLS search recognizes the following Grok patterns.

General Patterns

Pattern name	Regular expression
USERNAME	[a-zA-Z0-9._-]+
USER	%{USERNAME}
INT	(?:[+-]?(?:[0-9]+))
BASE10NUM	(?<![0-9.+-])(?>[+-]?(?:(?:[0-9]+(?:.[0-9]+)?)|(?:.[0-9]+)))
NUMBER	(?:%{BASE10NUM})
BASE16NUM	(?<![0-9A-Fa-f])(?:[+-]?(?:0x)?(?:[0-9A-Fa-f]+))
BASE16FLOAT	\b(?<![0-9A-Fa-f.])(?:[+-]?(?:0x)?(?:(?:[0-9A-Fa-f]+(?:.[0-9A-Fa-f]*)?)|(?:.[0-9A-Fa-f]+)))\b
POSINT	\b(?:[1-9][0-9]*)\b
NONNEGINT	\b(?:[0-9]+)\b
WORD	\b\w+\b
NOTSPACE	\S+
SPACE	\s*
DATA	.*?
GREEDYDATA	.*
QUOTEDSTRING	(?>(?<!\)(?>"(?>.|[^"]+)+"|""|(?>'(?>\\.|[^\\']+)+')|""|(?>`(?>.|[^`]+)+`|``))
UUID	[A-Fa-f0-9]{8}-(?:[A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}
DOMAINTLD	[a-zA-Z]+
EMAIL	%{NOTSPACE}@%{WORD}.%{DOMAINTLD}
QS	%{QUOTEDSTRING}

Networking-related Patterns

Pattern name	Regular expression
MAC	(?:%{CISCOMAC}|%{WINDOWSMAC}|%{COMMONMAC})
CISCOMAC	(?:(?:[A-Fa-f0-9]{4}.){2}[A-Fa-f0-9]{4})
WINDOWSMAC	(?:(?:[A-Fa-f0-9]{2}-){5}[A-Fa-f0-9]{2})
COMMONMAC	(?:(?:[A-Fa-f0-9]{2}:){5}[A-Fa-f0-9]{2})

Pattern name	Regular expression
IPV6	((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-Fa-f]{1,4}|((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-f]{1,4}){0,4}:((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]d|1dd|[1-9]?d)(.(25[0-5]|2[0-4]d|1dd|[1-9]?d)){3}))|:))(%.+)?
IPV4	(?<![0-9])(?:(?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2}))(?![0-9])
IP	(?:%{IPV6}|%{IPV4})
HOSTNAME	b(?:[0-9A-Za-z][0-9A-Za-z-]{0,62})(?:.(?:[0-9A-Za-z][0-9A-Za-z-]{0,62}))*(.?|b)
HOST	%{HOSTNAME}
IPORHOST	(?:%{HOSTNAME}|%{IP})
HOSTPORT	%{IPORHOST}:%{POSINT}

Path-related patterns

Pattern name	Regular expression
PATH	(?:%{UNIXPATH}|%{WINPATH})
UNIXPATH	(?>/(?>[w_%!$@:.,-]+|.)*)+
TTY	(?:/dev/(pts|tty([pq])?)(w+)?/?(?:[0-9]+))
WINPATH	(?>[A-Za-z]+:|\)(?:\[^\?*]*)+
URIPROTO	[A-Za-z]+(+[A-Za-z+]+)?
URIHOST	%{IPORHOST}(?::%{POSINT:port})?
URIPATH	(?:/[A-Za-z0-9$.+!*'(){},~:;=@#%_-]*)+
URIPARAM	?[A-Za-z0-9$.+!*'|(){},~@#%&/=:;_?-[]]*
URIPATHPARAM	%{URIPATH}(?:%{URIPARAM})?
URI	%{URIPROTO}://(?:%{USER}(?::[^@]*)?@)?(?:%{URIHOST})? (?:%{URIPATHPARAM})?

Date and time patterns

Pattern name	Regular expression
MONTH	b(?:Jan(?:uary)?|Feb(?:ruary)?|Mar(?:ch)?|Apr(?:il)?|May|Jun(?:e)?|Jul(?:y)?|Aug(?:ust)?|Sep(?:tember)?|Oct(?:ober)?|Nov(?:ember)?|Dec(?:ember)?)b

Pattern name	Regular expression
MONTHNUM	(?:0?[1-9]|1[0-2])
MONTHNUM2	(?:0[1-9]|1[0-2])
MONTHDAY	(?:(?:0[1-9])|(?:[12][0-9])|(?:3[01])|[1-9])
DAY	(?:Mon(?:day)?|Tue(?:sday)?|Wed(?:nesday)?|Thu(?:rsday)?|Fri(?:day)? |Sat(?:urday)?|Sun(?:day)?)
YEAR	(?>dd){1,2}
HOUR	(?:2[0123]|[01]?[0-9])
MINUTE	(?:[0-5][0-9])
SECOND	(?:(?:[0-5]?[0-9]|60)(?:[:.,][0-9]+)?)
TIME	(?!<[0-9])%{HOUR}:%{MINUTE}(?::%{SECOND})(?![0-9])
DATE_US	%{MONTHNUM}[/-]%{MONTHDAY}[/-]%{YEAR}
DATE_EU	%{MONTHDAY}[./-]%{MONTHNUM}[./-]%{YEAR}
ISO8601_TIMEZONE	(?:Z|[+-]%{HOUR}(?::?%{MINUTE}))
ISO8601_SECOND	(?:%{SECOND}|60)
TIMESTAMP_ISO8601	%{YEAR}-%{MONTHNUM}-%{MONTHDAY}[T]%{HOUR}:?%{MINUTE} (?::?%{SECOND})?%{ISO8601_TIMEZONE}?
DATE	%{DATE_US}|%{DATE_EU}
DATESTAMP	%{DATE}[-]%{TIME}
TZ	(?:[PMCE][SD]T|UTC)
DATESTAMP_RFC822	%{DAY} %{MONTH} %{MONTHDAY} %{YEAR} %{TIME} %{TZ}
DATESTAMP_RFC2822	%{DAY}, %{MONTHDAY} %{MONTH} %{YEAR} %{TIME} %{ISO8601_TIMEZONE}
DATESTAMP_OTHER	%{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{TZ} %{YEAR}
DATESTAMP_EVENTLOG	%{YEAR}%{MONTHNUM2}%{MONTHDAY}%{HOUR}%{MINUTE}%{SECOND}

Syslog patterns

Pattern name	Regular expression
SYSLOGTIMESTAMP	%{MONTH} +%{MONTHDAY} %{TIME}
PROG	(?:[w._/%-]+)
SYSLOGPROG	%{PROG:program}(?:[%{POSINT:pid}])?
SYSLOGFACILITY	<%{NONNEGINT:facility}.%{NONNEGINT:priority}>
HTTPDATE	%{MONTHDAY}/%{MONTH}/%{YEAR}:%{TIME} %{INT}
SYSLOGHOST	%{IPORHOST}

Log formats

Pattern name	Regular expression
SYSLOGBASE	%{SYSLOGTIMESTAMP:timestamp} (?:%{SYSLOGFACILITY})?%{SYSLOGHOST:logsource} %{SYSLOGPROG}:
COMMONAPACHELOG	%{IPORHOST:clientip} %{USER:ident} %{USER:auth} [%{HTTPDATE:timestamp}] “(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion})?|%{DATA:rawrequest})” %{NUMBER:response} (?:%{NUMBER:bytes}|-)
COMBINEDAPACHELOG	%{COMMONAPACHELOG} %{QS:referrer} %{QS:agent}

documentation@stormshield.eu