Proper Alerts Documentation

Release 1.2.2

Harsha

Jun 10, 2022

First Steps

Maintaining properly configured alerts is not an easy task in Splunk.

Is the search properly structured? Is data source still being indexed? Are search time range and schedule coordinated? What if the runtime is longer than the interval between one run to the next?

You probably answered these questions while doing the once in a while alert review.

This task however, becomes a pain in no time when the number of alerts rises and, as a consultant, it is common to encounter a Splunk environment in which admins lost track of configured alerts with consequences such as alerts being missed and improper resources usage.

A more efficient way to keep track of active alerts would consist in reviewing them all once, and then only review the ones that have been modified since their last review.

This the purpose of this App: help admins to continuously maintain properly configured alerts.

To do so, the App leverages Splunk KV Store to save active alerts in a lookup that gets updated every time an alert is modified. The lookup is then loaded into an interactive dashboard that lets Splunk admins review alerts.

CHAPTER 1

Topics

1.1 Overview

The App arbitrarily defines 6 *alerts checks* each alert should pass.

These checks are either *automatically* performed by *App's main report* or manually reviewed by Splunk admins through an *interactive dashboard*.

1.1.1 Alert checks

Check	Definition	Type
Source	Target data source must be indexed	Manual
Index	If applicable, target index(es) must be specified in the search query	Automatic
Runtime	Runtime must be lower than the interval between one run to the next	Automatic
Alignment	Alert schedule must be coordinated with search time range	Automatic
Delay	Alert must be scheduled with at least one minute of delay	Automatic
Structure	Search query must be correctly structured	Manual

Automatic checks

Index

When there is no index specified in a search query, Splunk searches in all available allowed indexes. This is not optimal in terms of resource usage and it is best practice to specify index(es) to be searched within the query. Searches that use alternate search commands in which index has not to be specified (e.g. `dbxquery`, `inputlookup`) are not taken into account (i.e. such queries are marked as having index specified). Resource

Note: Add any custom command to the *Search commands lookup* to have it considered as an alternate search command.

Runtime

When Splunk takes so much time to execute the query that search job is not finished when the alert's next run launches.

Alignment

Alert schedule must be coordinated with search time range. For instance, an alert running every 5 minutes should have a time range of 5 minutes to avoid duplicate alerts and for better use of resources. Resource

Note: Alignement check assumes that the interval between two alert runs remains even. While it should be the case to avoid overlapping, an uneven cron interval might be needed in some specific scenarios. This is not covered in this check just yet. In the mean time, it is possible to *whitelist a particular alert check*.

Delay

It is better practice to leave some delay on alerts by configuring a latest time of at least 1 minute. Resource

Manual checks

Source

Is there any data at all when you run alert's base search (i.e. query's first line)?

Structure

This a way more subjective check whose goal is to make sure search queries are properly written considering searches best practices. Resource

1.1.2 KV Store lookup

The `Update KV Store lookup` report is the core function of the App.

It checks for all enabled and scheduled alerts, perform the automatic checks and save results into a KV Store lookup.

See Update KV Store lookup report

1.1.3 Inventory dashboard

This dashboard loads KV Store lookup entries and lets Splunk admins review each alert independently.

During the review the admin will address alert manual checks and save results to the KV Store through interactive buttons.

See Review Alerts

1.1.4 Concurrency dashboard

The goal of this dashboard is to help resolve alert spreading issues.

Whith a growing number of alerts, there could be plenty of alerts launching at the same schedule.

This could be limited by the maximum concurrent scheduled searches Splunk scheduler can run.

Hence, the idea is to represent the number of alerts launched over time against this concurrency limit so it becomes easy to spot too busy schedules.

See Improve Spreading

1.1.5 Scheduler dashboard

This dashboard provides visibility over scheduler errors by showing the most frequent ones.

1.1.6 Runtime dashboard

This dashboard highlights the longer running scheduled searches.

1.1.7 Find dashboard

As Splunk admins we often need to find an active alert depending on various criteria such as its name, its recipient or what its search query contains.

The goal of this dashboard is to make this search easier.

1.2 Install

1.2.1 Prerequisites

These Apps must be deployed to your Search Head(s):

- Python Cron Iteration for Splunk
- Splunk App for Lookup File Editing

1.2.2 Deployment steps

1. Install the App on your Splunk Search Head(s)
2. Launch **Update KV Store lookup** from Reports tab by clicking `Open in Search`
3. Verify that active alerts are listed in the **Inventory** dashboard
4. [OPT] Adjust **getServiceRequest** macro to extract service request # from alerts' description
5. Set recipient to **Notify admin for alerts to review** alert or disable it
6. [WARN] Set recipient to **Notify alert recipient of a change** alert as `$result.email$` or disable it

Warning: **Notify alert recipient of a change** alert will send an email to alert's recipient when triggered.

1.2.3 Upgrade

Relaunch **Update KV Store lookup** from Reports tab by clicking `Open in Search`

1.3 Review Alerts

Alert review happens from the `Inventory` dashboard.

Active alerts should be listed in the very first panel:

reviewed	alert	app	owner	source	index	runtime	alignment	delay	structure	issues
✗	AllSplunkEnterpriseLevel - Core Dumps Disabled	SplunkAdmins	N/A	✗	✓	✓	✗	✗	✗	4
✗	AllSplunkEnterpriseLevel - Detect LDAP groups that no longer exist	SplunkAdmins	N/A	✗	✓	✓	✗	✗	✗	4
✗	AllSplunkEnterpriseLevel - Email Sending Failures	SplunkAdmins	N/A	✗	✓	✓	✗	✗	✗	4
✗	AllSplunkEnterpriseLevel - File integrity check failure	SplunkAdmins	N/A	✗	✓	✓	✗	✗	✗	4
✗	AllSplunkEnterpriseLevel - KVStore Process Terminated	SplunkAdmins	N/A	✗	✓	✓	✗	✗	✗	4
✗	Update KV Store lookup	Proper Alerts	N/A	✗	✓	✓	✗	✓	✗	3

Use the filters to narrow down displayed alerts.

The button is a reminder of *Alert checks definitions*.

Table info

column	description
reviewed	is the alert reviewed?
alert	alert name
app	alert app
owner	owner of the alert
source –> structure	is the check passed?
issues	# of failed checks

To review an alert, click on its row to display its specifics in a new panel:

Alert: **AllSplunkEnterpriseLevel - Email Sending Failures** (App: **SplunkAdmins**)

search	`comment("Find any failures to send emails due to either the size of the email or the email server not working or similar")` index=_internal `splunkenterprisehosts` "stderr from " python sendemail.py sourcetype=splunkd (`splunkadmins_splunkd_source`) | eval message=coalesce(message,event_message) | dedup message | rex "ssname=(?P<savedsearch>[^\"]+)" | rex "stderr from '[^']+':\s+(?P<error>.*)" | rex field=results_file ".*/dispatch/[^_]+__(?P<user>[^_]+)" | eval time=strftime(_time, "%+") | stats count, values(time) AS time by error, savedsearch, user | table time, count, error, savedsearch, user
cron schedule	3 * * * *
time range	earliest: -1h@h latest: now
last update	10/19/2020 18:18:54
last review	N/A
reviewer	N/A
app	SplunkAdmins
actions	
description	Chance the alert requires action? High. Ideally this action shouldn't be using email but this should fire when the email server is throwing errors
owner	N/A
service request	N/A
alert checks	source: ✗ index: ✓ runtime: ✓ alignment: ✗ delay: ✗ structure: ✗

The `Review alert` section underneath provides interactive buttons:

alert's search query in a new tab

alert actions from scheduler logs in a dynamic panel

edit the alert in its App context in a new tab

reload results

Reloading results

If you have just edited the alert - to specify an `index` for instance - and you want the results to be refreshed right away, click the button as it launches the `Update KV Store lookup` alert in the background.

Whether automatic checks are passed or not, you can then update *manual checks definitions* from the `Update data` section.

To do so, update each manual check status by clicking either on ✓ or .

Update buttons

- If it is currently failed and you want to review it as passed, click <check> ✓
- If it is currently passed and you want to review it as failed, click <check>
- If you want to mark it as reviewed, click Reviewed ✓
- The same applies for the `service request` reference

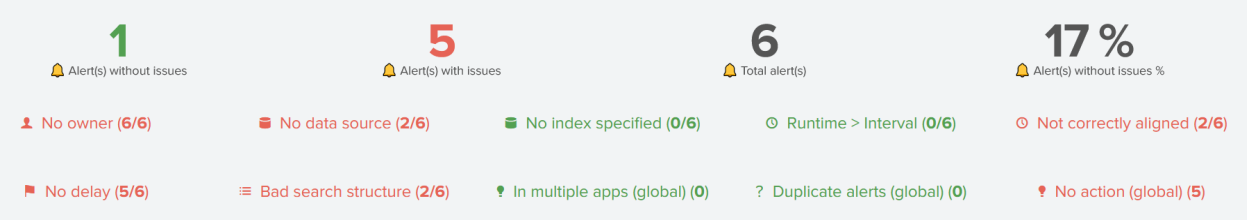
Note: Whatever manual check updated, current Splunk admin becomes alert's `reviewer`.

1.4 Browse KPIs

1.4.1 Alert checks stats

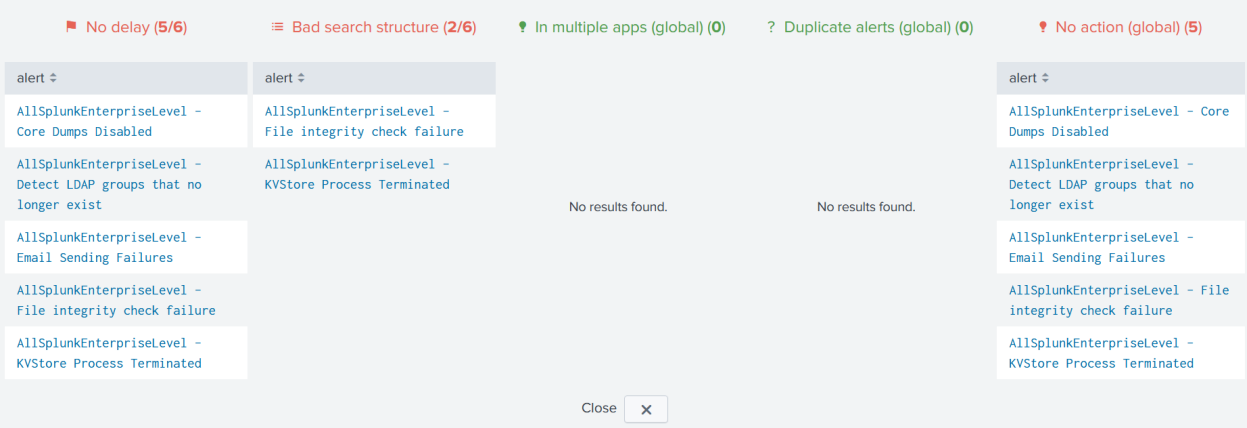
`Issues` dashboard is stats oriented.

Its purpose is to ease alert reviewing by tracking down its progression:



Stats can be shown by app, owner or even status, reviewed or not.

Click on a panel title to obtain the list of alerts related to each stat:



Use the `Close` button to hide the panel back.

1.4.2 Additional checks

Besides stats on defined *Alert checks*, 3 additional checks are provided:

Check	Description
In multiple apps	The same alert is configured in multiple Apps
Duplicate alerts	Several alerts share the same search query, schedule time and time range
No action	Alert has no configured action
Close names	Alerts with close names

In multiple apps

This is not a big issue since Splunk merges the parameters across different Apps but it can lead to confusion when editing the alert so it should be fixed.

Duplicate alerts

As it might indicate a duplicate alert - same alert, different name - it should be checked.

No action

This check discards alerts whose action resides within the search query (e.g. outputlookup). These alerts should be checked.

Close names

This check's purpose is to spot alert having very close names to check for duplicate alerts.

Warning: These additonal checks are global, meaning dashboard's top filters does not apply.

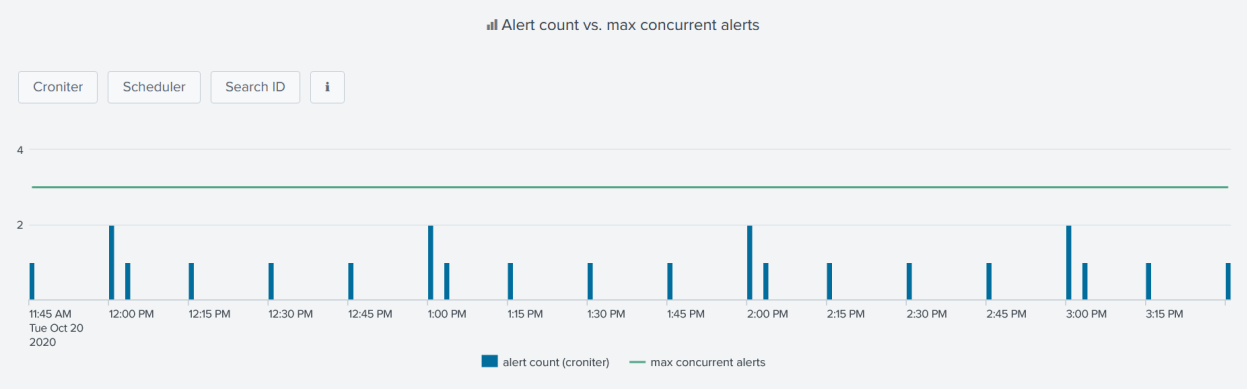
Tip: Click on an alert row to open it in the `Inventory` dashboard.

1.5 Improve Spreading

Spreading issues are addressed in the *Concurrency dashboard*.

1.5.1 Concurrency limit

The first panel shows the number of alerts scheduled over time against the maximum number of concurrent shcedule searches that Splunk scheduler can run:



The idea is to spot too busy schedules and to better spread alerts so that concurrency limit does not get maxed out.

There are 3 versions of the time chart, each having his own source for determining the number of alert running every minute.

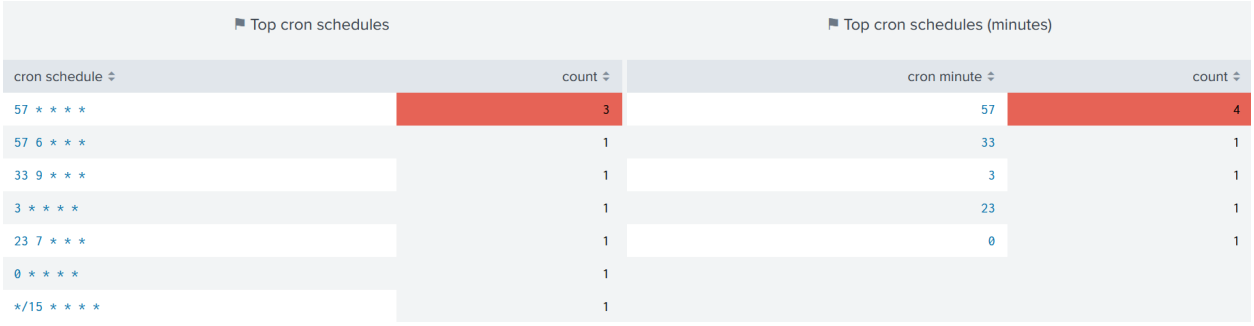
Additional info is provided within the dashboard accessible from the button:

Time chart	Data source
Croniter	Python Cron Iteration for Splunk convert each alert schedule to a timestamp
Scheduler	Scheduled search log events
Search ID	Search IDs events

Tip: As Croniter converts alerts' cron schedules in timestamps, any alert adjustment would be reflected right after the panel is refreshed.

1.5.2 Busiest schedules

Additional panels underneath have the same purpose of spotting busy schedules by showing the most used ones:



Tip: Clicking on a cron schedule would open a new panel with matching alerts which can then be clicked again to open alert's edit page.

1.6 Lookups

This App uses 4 lookups, `alerts_lookup`, `search_commands_lookup`, `alert_whitelist_lookup` and `app_whitelist_lookup`

All these lookups can be edited manually effortlessly from the `Lookups` tab which relies on Splunk App for Lookup File Editing.

1.6.1 KV Store lookup

`alert_lookup` is the KV Store lookup that store the state of active alerts.

It has the following fields:

Field	Type
actions	string
alert	string
app	string
app_label	string
cron_schedule	string
dataSourceExist	bool
dateLastReview	number
description	string
earliest_time	string
email	string
hasOwner	bool
hasServiceRequest	bool
indexIsSpecified	bool
interval	number
latest_time	string
md5	string
md5_search	string
owner	string
reviewer	string
run_time	number
runtimeIsLowerThanInterval	bool
scheduleHasAtLeastOneMinuteDelay	bool
search	string
searchIsCorrectlyStructured	bool
searchPeriodIsAlignedWithSchedule	bool
service_request	string
updated	number

1.6.2 Search commands lookup

`search_commands_lookup` is the lookup that store splunk search commands.

It goes like this:

command	classic_search	output_command
<Splunk command>	bool	bool

It is used in the *indexIsSpecified* macro which checks if the index is specified in alerts' queries.

It is also used in the `No action` *additional check* from the `Issues` dashboard which looks for alert without any configured action.

1.6.3 Whitelists

You might want to whitelist an alert's specific check or even all alerts for a given App.

This is possible using whitelisting lookups.

Hint: In both whitelists below, the App name to use is the one from App's URL `../en-US/app/<app_name>/..`

Alert whitelist lookup

Use this lookup if you want to whitelist a specific check for a given alert.

alert	app	index	runtime	alignment	delay
<alert name>	<app name>	bool	bool	bool	bool

To whitelist an alert's specific checks, add alert's name and app, and set the check to be whitelisted to `1`.

In the example below, alert `foo` from the `bar` App has both its index and alignment checks whitelisted:

alert	app	index	runtime	alignment	delay
`foo`	`bar`	1	0	1	0

Note: The whitelisted checks will be considered as ✓ in both `Inventory` and `Issues` dashboards.

App whitelist lookup

Use this lookup if you want to whitelist an entire App from being checked.

app
<app name>

To whitelist an entire App, just add its name to the lookup.

Note: The whitelisted Apps will not be considered at all in both `Inventory` and `Issues` dashboards.

1.7 Reports

1.7.1 Update KV Store lookup

This reports looks for all active alerts, perform the automatic checks against each and save results into a KV Store lookup.

Lets break down its search query:

Search for active alerts

1-2	Search for all enabled and scheduled alerts, then for each alert:
3	Add *triggeredalerts*' to `actions` if `alert.track` is true
4	Set recipient field only if `action.email` is true
6	Check if the index is specified in the search query, if applicable, using macro `indexIsSpecified`
8-9	Extract service request reference from the description field, using macro `getServiceRequest`
11	Clean `updated` field
12-15	Set fields to `N/A` so that MD5 hash is never empty
16	Save the MD5 hash of the concatenation of main fields for later comparison

Considered fields

Field	Field Description
`alert`	alert name
`app`	app name
`updated`	last update timestamp
`cron_schedule`	alert schedule
`cron_schedule`	alert schedule
`earliest_time`	search period earliest time
`latest_time`	search period latest time
`search`	search query
`email`	recipient(s)
`actions`	alert action(s)
`owner`	alert owner (knowledge object)

Also save the MD5 hash of the search query.

17-19	Use Cron Iteration command to calculate the interval between 2 executions
20	Calculate the search time range interval using earliest and latest time
21	If search time range interval = cron schedule interval, or if query is not an `index=foo` search, *Alignment* check is passed
23-24	Prefix all fields name except `alert` & `app` with `new_` for later comparison
25-35	Determine the maximum runtime from scheduler logs
36	Filter out alerts only present in scheduler logs

Compare it to current KV Store lookup entries

37-38	Add the current content of the KV Store lookup to the results for comparison
39	Group both data sets (1-2 & 30-31) by `alert` and by `app`
40	If the MD5 of main fields have changed or if runtime exceeds interval, or if indexIsSpecified value differs, keep the newest values
41	If the search query has changed, reset the *Structure* check
42	If the search query has changed, reset the *Source* check
44	If the runtime exceeds the interval, update the *Runtime* check
46-47	Check if the search period has a minimum delay of 1 minute, if applicable
48-51	Fields clean up
52-55	Retrieve App label

Save results to the KV Store lookup

56	Call KV Store lookup to get the `_key` field for each entry to update
57	Update `alert_lookup` KV store lookup entries with the results

The output can be both:

- alerts created after the last run of the report
- alerts modified since the last run of the report

Warning: Report runs every hour. If you change its cron schedule to your needs, adjust time range accordingly.

1.7.2 Clean KV Store lookup

Whenever an alert is enabled and scheduled, it is saved it the KV Store lookup thanks to the `Update KV Store lookup` report above.

If the same alert is disabled or even deleted later on, it has to be removed from the KV Store lookup.

This is what the `Clean KV Store lookup` report does, removing disabled or deleted alerts from the KV Store lookup.

1	Load the current content of the KV Store lookup
2	Mark this data set with the key value `source=kv_store`
3-7	Append the list of enabled and scheduled alerts marked with key value `source=rest`
8	Group both data sets (1-2 & 3-7) by `alert` and by `app`
9	Count the number of data sets each alert is in
10	Filter out alerts that are only part of 1 data set
12	Save results to the KV store

Hint: In simple steps, the report loads all entries from the KV Store lookup, takes out disabled or deleted alerts, and overwrites the output back to the lookup. As a result, deleted or disabled alerts are no longer in the lookup.

1.8 Alerts

1.8.1 Notify admin for alerts to review

This alert notifies Splunk admins of the count of alerts that need to be reviewed.

The idea is to enable it after the first initial review of all alerts.

This way, Splunk admins get notified of any alert to review whether new of modified.

The recipient(s) must be set and the schedule should be adjusted to your needs.

Email body contains the following message:

```
There are <count> remaining alerts to review.
```

1.8.2 Notify alert recipient of a change

This alert notifies the recipient of an alert of any change made on an alert is the recipient to.

The goal is to avoid any issue that could arise from unsolicited or unannounced modifications.

The recipient of this alert is the recipient of the modified alert.

Note: If the alert has no recipient, alert is sent to email set in `Notify admin for alerts to review` alert.

Email body contains the following message:

```
Your alert '<alert name>' has been modified.
Please find below what has changed - prefixed with new - within alert's main␣
↪parameters.
```

It also comes with the inline table below:

modification date	alert	app	`<field>`	new `<field>`

Note: Possible `<field>` values: `cron schedule`, `earliest time`, `latest time`, `search`, `actions`, `email`, `owner`

Attention: `new <field>` column comes up only if there is a new value for the said field. If the `new <field>` value is `N/A`, please do not consider. The column shows up because there was a new value for that field in another modified alert triggered at the same time.

Search query steps:

1	Search for all enabled and scheduled alerts, then for each alert:
3	Add triggeredalerts‘ to actions if alert.track is true
4	Set recipient field only if action.email is true
7	Clean `updated` field
8-11	Set fields to N/A so that MD5 hash is never empty
12	Save the MD5 hash of the concatenation of *main fields* for later comparison
13	Clean `latest_time` field
14	Prefix all fields name except `alert` & `app` with `new_` for later comparison
16-19	Load KV Store lookup entries that do have an `owner`
20	Group both data sets (1 & 9-12) by `alert` and by `app`
21	Filter out results having the same MD5 hash of main fields in both data sets
24-29	Eval main alert fields to identify the modified ones
35-38	Retrieve App label
42-49	If email is invalid set it as set in `Notify admin for alerts to review` alert
51	Fill any `null` column with `N/A`

1.9 Macros

1.9.1 indexIsSpecified

This macro is used in `Update KV Store lookup` report to perform the *Index automatic check*

It looks through the `qualifiedSearch` field of each alert to find if an index is specified in the query.

While sounding simple at first, several usescases has to be covered:

- subsearches
- macros
- eventypes
- alternative search commands

As a result, the underlying query handles:

- up to 2 levels of macros
- up to 2 levels of eventtypes
- alternative search commands via the `search_commands_lookup`

This latter case is simple, if the search is not a classic `index=foo` search, then index is not required.

While it still is a good practice to precise the index using alernate commands such as mstats, the initial goal is to spot uneficient search queries within alerts, so mainly classic searches.

If you use a scripted command in an alert query and it is spooted as not having index specified, add your script command to the `search_commands_lookup` and give the value `classic_search = 0`

Note: This macro does its best to spot alert queries in which no index is specified. While it tries to cover most cases, false positives are always possible. In this case, *please report* so it can be improved. You can also *whitelist* the index check for any given alert.

1.9.2 getServiceRequest

This macro is used in `Update KV Store lookup` report to perform extract the service request from alert's description field.

This a common practice to add service request reference(s) to alert's description.

Having service request could be useful when investigating an alert from the inventory dashboard as a way to quiclky find context, requesstor or any interesting details before doing any chnages

1.9.3 maxConcurrentSavedSearches

This macro is used in the `Concurrency` dashboard to retrieve the maximum number of concurent saved searches that can be run.

1.10 Download

Download the App from:

Splunkbase

GitHub

1.11 Compatibility

1.11.1 Splunk

7.x	✓
8.x	✓

1.12 Release Notes

Version	Date	Comments
1.2.2	19 May 2022	Fixed issues in 'Find' dashboard
1.2.1	5 May 2022	Added new dashboards: Scheduler, Runtime, Find Updated icons in all dashboards Removed the update service request function from Inventory dashboard Granted r/w access for Splunk Cloud role sc_admin
1.1.5	17 Jan 2021	Saved searches can now be searched by type - alert or report
1.1.4	12 Nov 2020	Fixed whitelisting alert issue in *Inventory* dashboard Fixed *IndexIsSpecified* macro
1.1.3	9 Nov 2020	Fixed *No action* check in *Issues* dashboard Added *Close names* check in *Issues* dashboard Fixed *Update KV Store lookup* query: - recipient field is set only if `action.email` is true - `triggeredalerts` is added to `actions` when `alert.track` is true - null fields are set to `N/A` so that MD5 hash is never empty
1.1.2	8 Nov 2020	Alert owner is now the owner of the knowledge object instead of the first recipient of the alert Index check has been improved Alerts and Apps can now be whitelisted from the checks `Notify alert recipient of a change` has be renamed to `Notify alert recipient of a change` and improved
1.1.0	30 Oct 2020	Fixed `Notify alert owner of a change` alert
1.0.7	22 Oct 2020	Initial release

1.13 Support

App is community supported through the following options:

1.13.1 Splunk Answers

Start a conversation on Splunk Answers tagged `Proper Alerts`

1.13.2 GitHub

Create an issue on GitHub

1.13.3 Email

alh-spk@protonmail.com