DRAFT: Performance Insights for Splunk

Diagnosing and Troubleshooting Performance Issues

June, 2025

Overview

This document details the process, with examples, for using the Performance Insights for Splunk (PerfInsights) app to diagnose and discover potential performance issues with Splunk Enterprise deployments and premium applications. PerfInsights is a tool that can be installed on both Splunk Cloud and Customer Managed Platform (CMP) deployments. It uses internal indexes to gather and calculate key metrics that give a good overview of system stability and performance.

PerfInsights does not collect any new data, but relies on data that is already being collected by Splunk itself. This means that there is no performance overhead in installing the tool in your environment.

PerfInsights will use additional resources during an active investigation: The dashboards provided in the tool generate additional searches. The amount of resources used depend on the length of the investigation, and the time span of indexed data over which the investigators search. In general, the dashboards are designed to minimize the search overhead by making use of indexed field equality operators and search result re-use.

Perf Insights is not an active health monitoring app. It will not generate notifications, or make assumptions about the health of your deployment. The metrics it provides are open to interpretation by the user, since what might be a potential problem in one environment might be perfectly normal in a slightly different one.

Installation

PerfInsights is available on Splunkbase (https://splunkbase.splunk.com/app/5549). To install PerfInsights, access your Splunk Enterprise UI as an administrator and Select Apps->Find More Apps. Select “IoT & Industrial Data” and search for “Performance Insights”. Locate the application and install it.

Note: Once installed, you will need to wait for the application to be replicated to the search heads before it is fully functional. You may see search errors if you try to use it before it is ready. If this happens, wait a few minutes and try again.

Diagnostic Process

There are two ways that PerfInsights is typically used to diagnose performance issues: a reactive approach, and a proactive approach.

Reactive Diagnosis

Reactive diagnosis, as the name suggests, is when a problem has already occurred and the goal is to discover what caused the problem. In these cases, the actual behavior of the system, from a user’s point of view, has deviated from the expected behavior. An example of a deviation might be that users’ searches report warnings that some data might be missing.

To conduct a reactive diagnosis:

1) Focus the tool dashboards around the point in time when the issue first appeared,
2) Observe what was happening in the system at that time to see what may have caused the unexpected behavior. Things to look for include:
    a) High/saturated resources (e.g., CPU, memory, Network IO, Disk IO)
    b) High/saturated queues (e.g., search or indexing queues, replication queues)
    c) Plateaus (e.g., search or indexing throughput)
    d) Sudden spikes (e.g., search concurrency, error rates)
    e) Anything that looks unusual.
3) Separate cause and effect. Adjust the time range to determine the order of abnormal events.
4) Investigate the abnormal events starting from the earliest.

For each abnormal event, the goal is to determine if it is a contributing cause, a symptom, or an unrelated event. Determining this will often require additional searches or tools beyond PerfInsights.

Proactive Diagnosis

Proactive diagnosis involves using PerfInsights to look at system metrics in an attempt to find trends that could lead to issues in the future. As an example, if you look at search times, and notice that they are increasing over time, this could lead to an increase in search concurrency and eventually skipped searches.

Note: Proactive diagnosis is subject to the Observer Effect, in which the act of observing the system affects the system behavior.

- Be cautious when observing unstable systems as the extra search load could further destabilize the system, and
- When reading PerfInsights charts and tables, be sure to take into account the search load the tool is adding.

To conduct a proactive diagnosis:

1) Start with a large time range, ending at the current time (e.g., past 7 days to now)
    a) Look for charts that are trending upwards (e.g., CPU, memory, search times, queue lengths)
    b) Look for spikes in charts that last longer, or become more frequent (e.g., search concurrency, bundle replication)
2) Repeat with smaller time ranges. (e.g., Past 1 day or 1 hour). While long time ranges can show trends, it can also hide them. Shrinking the time range allows you to see more granular data. Spikes that might have been aggregated away over the span of a week, might show in the span of an hour.

For each potentially problematic behavior observed, attempt to understand the cause. For example, if you observe that search times are increasing over time, you might also find that your data ingestion rate has also been increasing. Determining if this will be an issue depends on the current state of the system, and any further increases in data ingestion rates.

Deeper Dives

The intention behind PerfInsights is to provide a single application that can help identify all potential performance issues, and to provide as much detail as possible to help resolve those issues. However, in some cases, PerfInsights does not, or cannot provide an adequate amount of information to complete the performance analysis. When this is the case, use this tool as a jumping off point for further investigation.

Expand Existing Charts

Sometimes the existing charts come close to showing what you need, but require some tweaking. Using the magnifying glass link on a chart will open the search in a new window, allowing for that customization. All the dashboards are editable, so if you find yourself making these customizations a lot, you can modify the search directly in the dashboard, or add a new chart to the dashboards. Be aware that reinstalling the

application will overwrite your changes. If your change could benefit others, consider sending a message to the support team with the suggestion.

Add System Data to Indexes

By default not all OS level logging is available in indexes. This information can be very useful when investigating performance issues. On CMP deployments, consider adding local forwarders on your servers configured to read system logs and index them.

Forming and Testing Hypotheses

Performance issues are often complex and multi-faceted; determining root causes can be tricky, and even sound and logical reasoning can lead to incorrect conclusions when not all the data is accounted for. Resolving issues can be an iterative process.

Once you've observed the situation, and come up with a reasonable explanation for the behavior, it's time to test that hypothesis. Here, even if you started in a reactive diagnosis scenario, you will be iterating over proactive diagnoses.

For each iteration:

1) Start by identifying a minimal set of changes that could correct the issue.
    - Use Splunk Documentation, Splunk Support, or the Splunk Community.
2) Change only one thing at a time, if possible.
3) Using the same charts that allowed you to build your hypothesis, verify a positive change over similar time periods that exposed the issue.

Troubleshooting

Symptom	Possible Causes	Diagnose	Remediate
Slow Search Times	High Concurrency: Any search load above the system capacity necessarily waits, leading to increased search times.	Look for search concurrency values above the single indexer CPU count. Both spikes or consistent values above the system capacity are problems. Go to the Search Concurrency section to see if this applies to your situation.	Flatten search load by identifying spikes in scheduled searches; these often occur at a regular cadence (e.g., every 15 minutes, the top of the hour, or midnight). Where possible, move scheduled searches to less busy times. Make use of scheduled search windowing or skewing to spread search start times over larger ranges. This blog discusses the topic further: Schedule Windows vs. Skewing | Splunk
	Inefficient SPL: Slow searches can lead to higher search concurrency.	Using the search performance metrics, look for searches with a large total run time (a combination of run times and frequency). If the total run time for a search is a significant percentage of the selected time range (e.g., 360s over a 1 hour range, or 10%), this is a good candidate improvement. Go to the Search Metrics section to see if this applies to your situation.	Tips for improving search SPL: • Decrease time ranges. • Use only necessary indexes. • Filter out as much data as possible before non-streaming or transforming operations. • Filter on indexed fields and use the **::** operator. • Use the **TERM()** operator where possible for text searches. • Use **tstats** instead of stats if you don't need the raw data. Useful links on this topic:

			Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk Writing better queries in Splunk Search Processing Language Use CASE() and TERM() to match phrases - Splunk Documentation
	Too Many Buckets: Searches that span large time ranges need to open/inspect many buckets, which can be inefficient.	Check your bucket upload/download or disk read/write rates. If the network transfer rates or disk IO are significant or cache hit rates are low you may benefit from fewer buckets. Go to the Buckets or Resource Monitoring sections to see if this applies to your situation.	Increase the maxDataSize setting in indexes.conf to a MB value larger than your typical warm bucket size. This is an index specific setting. More information on this can be found here: indexes.conf - Splunk Documentation
	Not Enough Buckets: Large bucket writes can be slow and the low number of files means less parallelization and higher latency on those writes.	Check your bucket upload/download or disk read/write rates. If the network transfer rates or disk IO are not significant and cache hit rates are high you may benefit from more buckets. Go to the Buckets section to see if this applies to your situation.	Decrease the maxDataSize setting in indexes.conf to a MB value smaller than your typical warm bucket size. This is an index specific setting. More information on this can be found here: indexes.conf - Splunk Documentation
	Connectivity Issues: Unstable network IO from saturation or latency issues can lead to excessive retries.	Scan errors and warnings looking for connectivity issues or communication retries. Go to the Environment Diagnose section to see if this applies to your situation.	For any error or warning that looks like a network timeout with a retry, determine the cause of the error. Check your network throughput and latency, and ensure you have adequate bandwidth, Check packet routing between systems to ensure data is not flowing through unexpected paths or being held up by unnecessary network rules.
	Undersized Deployment: When search concurrency is not exceeding system capacity, but there is high CPU usage on indexers or search heads, the deployment may be too small.	Check for CPU usage over time and compare with search load. When not under search pressure, CPU usage should be low. At peak search load, CPU usage should be under 80%. Go to the Search Resource Usage, Search Concurrency, and Search Metrics section sto see if this applies to your situation.	The best practice for sizing a deployment : Set up a test environment using a representative subset (e.g., 1/10th) of your ingest data, then size your search heads and indexers to run optimally at peak search load (no more than 80% CPU, and all other resources stable). Your optimal deployment will use the exact same search head configuration, but with indexers scaled linearly to handle the full ingest data set (e.g., 10x larger ingest deployment). If a test environment is not an option, then add resources as needed. Note that searches put load on both indexers and search heads, while ingest generally only puts load on indexers. Also note that if your concurrent search counts exceed your single indexer core count, you will likely benefit from fewer indexers of a larger CPU count. For example, four 32 core indexers will outperform

			eight 16 core indexers.
Skipped Searches	"Maximum Number of a Class of Search Types (e.g., Historical) Reached": This indicates that your system is trying to process too many searches at the same time.	Look for searches that are scheduled for the same time. Look for searches that are scheduled to run shortly after searches with long durations. Look for scheduled searches with long durations. Go to the Search Skip Details and Slow Search Times sections to see if this applies to your situation.	Reducing search runtimes will help reduce search concurrency. Make sure long-running searches are optimized. Spread scheduled searches out to use less busy times. Note: Do not increase search concurrency limits. While this might be tempting in order to make the problem go away. The effects are temporary and will likely make the situation worse. For more information: Schedule Windows vs. Skewing | Splunk Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk Writing better queries in Splunk Search Processing Language Use CASE() and TERM() to match phrases - Splunk Documentation
	"Maximum Number of This Search Reached": This usually means a scheduled search ran longer than its scheduled interval.	Look for scheduled searches for which runtimes come close to, or exceed the interval between runs. Go to the Search Skip Details and Slow Search Times sections to see if this applies to your situation.	See cell above.
Unexpected Search Head Restarts or Unavailability	Out of Memory: The most frequent cause of search head restarts is memory pressure. While out-of-the-box system default limits are configured to minimize memory issues, poorly designed searches run under service accounts can exhaust memory quickly.	Look for searches that require a lot of search head memory. Go to the Search Resource Usage section to see if this applies to your situation.	Reduce result sets in searches to only the necessary data. Make use of summary indexing or tstats. Front-load streaming commands to put more load on the indexers. See these links for more information: Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk Writing better queries in Splunk Search Processing Language
	CPU: If server load causes CPU saturation, the servers may fail to respond to health checks in a timely manner. The search head cluster captain will stop sending jobs to that search head, which could overload the remaining search heads.	Look for periods where fewer than the expected number of search heads are running. Look for a cadence of spikes in search load. Look for long-running ad hoc searches. Go to the Search Resource Usage and Running Search Heads sections to see if this applies to your situation.	Smooth out search spikes by moving scheduled searches to less busy times. Set user's disk and search quota limits to avoid excess CPU load. See these links for more information: Schedule reports - Splunk Documentation Use cron expressions for alert scheduling - Splunk Documentation Authorize.conf

Ad Hoc Search Errors	Timeouts: With most SPL searches, for every efficient way to perform a search, there are many inefficient ways. Inefficiencies lead to long running searches, which may hit timeouts.	Look for excessive concurrent searches leading to long search queues. Look for logs indicating that searches failed to complete. Go to the Search Metrics and Environment Diagnose sections to see if this applies to your situation.	Tips for improving search SPL: • Decrease time ranges. • Use only necessary indexes. • Filter out as much data as possible before non-streaming or transforming operations. • Filter on indexed fields and use the **::** operator. • Use the **TERM()** operator where possible for text searches. • Use **tstats** instead of stats if you don't need the raw data. Useful links on this topic: Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk Writing better queries in Splunk Search Processing Language Use CASE() and TERM() to match phrases - Splunk Documentation
	Incomplete Results: Incomplete results are often due to a part of the search (e.g., subsearch) failing, or an unexpected restart of an underlying server. Note: Incomplete results can also occur when join or subsearch limits are reached. Increasing these limits can have negative performance effects.	Look for exhausted system resources. Look for unstable systems and server restarts. Check logs for errors indicating data truncation. Go to the Resource Monitoring, Environment Diagnose, and Search Metrics sections to see if this applies to your situation.	Reduce result sets in searches to only the necessary data. Make use of summary indexing or tstats. Front-load streaming commands to put more load on the indexers. Ensure your deployment is sized correctly. Note: The cause of server instability may exist outside of your Splunk configuration. If you suspect this, please involve your Site Reliability Engineers. See these links for more information: Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk Writing better queries in Splunk Search Processing Language
Unexpected Search Results	Bundle Replication Lag: Large bundles (large lookups with frequent changes), and network saturation can lead to slow or missed bundle replications. This can, in turn, lead to the use of stale data from lookups.	Look for bundle replication times consistently higher than 100ms. Look for bundle replication delta sizes consistently over 10KB. Go to the Bundle Replication section to see if this applies to your situation.	Ensure changes to large lookup files are infrequent. Use a larger number of smaller lookups rather than monolithic all-in-one lookups. Ensure network traffic bandwidth is adequate. For more information on troubleshooting bundle replication look here: Troubleshoot knowledge bundle replication - Splunk Documentation

	Configured Search Limits Reached: Out-of-the-box system default limits are configured to reduce resource saturation. These limits can cause subsearches to return fewer than expected results.	Look for errors and warnings returned to the end user in the search UI.	Ensure subsearches return a reasonably small amount of data. Where necessary, increase subsearch limits. Note: saved searches, subsearches, joins, and commands like top may all have their own limit settings.
Uneven Search Head Server Load	Captain Limits: If one search head differs from the rest, this may be the captain, and that is expected.	Look for a single search head that is processing fewer searches. If only one shows lower search counts, and the search run times are not drastically different for that server, there is likely no need for any further action. You can confirm the search head captain using this search: `| rest /services/shcluster/status splunk_server=local | fields captain.label` Go to the Search Counts section to see if this applies to your situation.	No Action.
Uneven Server Load (any server)	Unhealthy Server: A server that is stuck in a restart loop, or suffering heavy CPU usage will likely see lower search counts and higher average search runtimes than other servers.	Look for servers (other than the search head captain) that are using far more or far fewer resources (CPU, Memory, Network, Disk) than the others. Check logs for any server reporting errors, or servers that have no error logs at all. Go to the Resource Monitoring and Environment Diagnose sections to see if this applies to your situation.	Identify causes of the problematic server behavior, either through Splunk error logs, external server diagnostic tools, or by checking server configurations. Once the issue has been corrected, reduce load temporarily, then restart the problematic server. Reducing load is important so that restarting servers does not cause excess load on the remaining servers, potentUse the CLI to view information about a search head cluster - Splunk Documentationially putting them in a bad state.
	Load Balancer Configuration: If the load balancer has the incorrect information about the cluster servers or their statuses their load may be routed incorrectly.	Look for servers (other than the search head captain) that are using far more or far fewer resources (CPU, Memory, Network, Disk) than the others. Ensure all servers are healthy with no obvious error. Go to the Resource Monitoring and Environment Diagnose sections to see if this applies to your situation.	If you find search heads that are suspect, you can confirm the cluster status using the CLI (Use the CLI to view information about a search head cluster - Splunk Documentation), or simply restart the search heads through the UI. This often corrects any search head issues. If you find indexers that are suspect, restarting them is usually the best option. Restart the entire indexer cluster or a single peer node - Splunk Documentation Be sure to reduce load on your system before restarting servers.
	Load Balancer Bypassing: If forwarders or inter-server communication has been	Look for servers (other than the search head captain) that are using far more or far fewer resources (CPU, Memory,	Scan your Splunk configuration files looking for hard-coded IP addresses or server names. Ensure those

	configured with IP addresses or hard-coded machine names, machine load may be unbalanced.	Network, Disk) than the others. Ensure all servers are healthy with no obvious error. Go to the Resource Monitoring and Environment Diagnose sections to see if this applies to your situation.	addresses and names refer to load balancing systems or have a valid reason to be hard-coded. List of configuration files - Splunk Documentation Look for forwarders that are sending to a subset of the stacks, or to just one site.
System Hangs	Faulty subsystem on an indexer, leading to slow or hanging search results. This can lead to a number of searches getting stuck over time, and can halt the system.	Look for sudden increases in search concurrency, and deployment-averaged indexer memory, coupled with a decrease in deployment-averaged search head and indexer CPU activity. A single indexer with poor storage or network performance can degrade the entire cluster if there are no other health issues. Go to the Search Concurrency and Resource Monitoring sections to see if this applies to your situation.	Identify any indexer that is performing sub-optimally in terms of IO. It will usually be the first one to show a drop in CPU usage before the system performance degrades entirely. Stop the affected indexer, correct the IO issues, and restart the indexer.

Diagnoses

Performance Trend

The performance trend section is a good starting place to get a sense of system activity, though it is often too high-level to determine the root cause of any issue.

Event Ingestion Volume

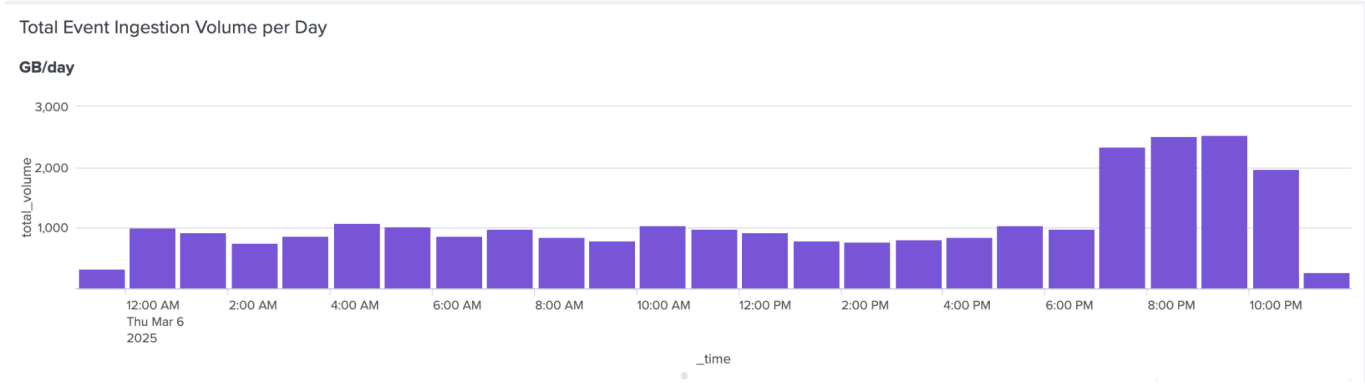
What it Indicates

By default this chart looks back at the ingestion volume over the past week, grouped by day. Knowing when and how much data is flowing into your system allows for proper sizing and tuning to handle peak load.

What to Look For

Spikes:

When looking at a long time range, aggregated over a small number of groupings, look for groups that deviate. For example, looking at 1 week's worth of data, if one day is larger than the others, narrow in on that day. Spikes can cause performance problems if resources are already running close to capacity.



An increase in Ingestion Rate will likely cause performance changes elsewhere in the system.

If you do see spikes, first be sure the data is expected. If it is expected and you are experiencing issues

starting at that point, determine if the incoming data can be reduced or broken into smaller chunks spread over longer time periods.

Troughs:

Similar to spikes, troughs can indicate an unexpected change in data flow. Troughs are unlikely to cause performance issues, but should be investigated to ensure data is not missing.

Average Search Runtimes

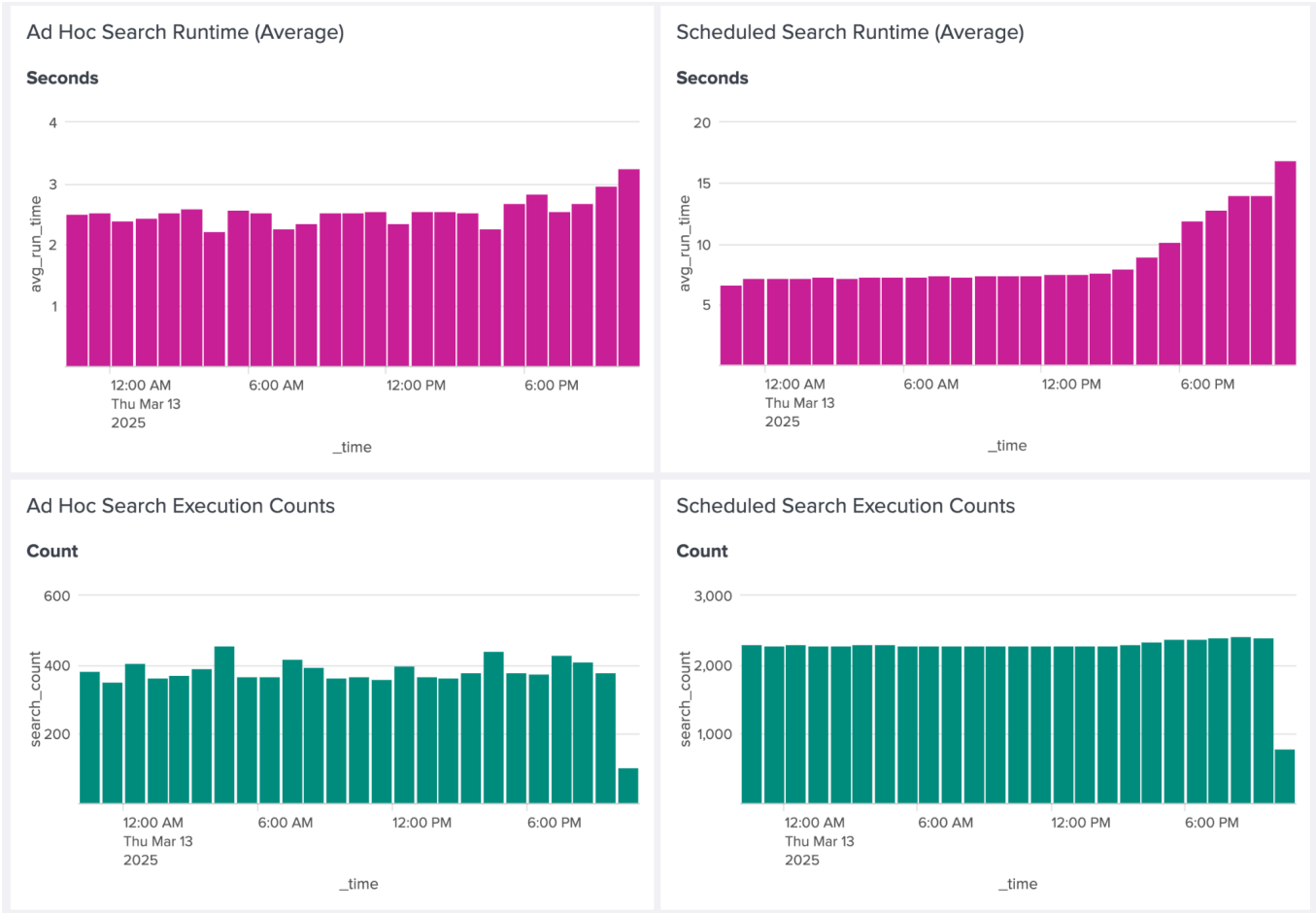
What it Indicates

By default these charts look back at average search times over the past week, grouped by day. Search load will likely have some period over which load and type is somewhat consistent, and in that period average search times should also be consistent.

What to Look For

Gradual Increases:

Given consistent search load and ingest volume, search runtimes should also be consistent. While some variability is expected, any noticeable increase in runtimes over time is cause for concern. Left unchecked, searches will eventually timeout or exhaust resources or queues.



Over long periods of time, the charts should remain flat. Upward trends indicate a potential problem.

When search runtimes increase over time, use the Search Metrics section below to determine which searches are taking longer, and act accordingly. If all searches are taking longer look for increases in search concurrency (Search Metrics section), or exhaustion of resources (Resource Monitoring section).

Search Execution Counts

What it Indicates

Search execution counts give an indication of how much user activity there is, or how your scheduled searches are distributed. Scheduled searches will likely be nearly equal from day to day, so variance is caused by user-driven ad hoc searches.

What to Look For

Looking back over days, weeks, or even years can give you a sense of when your system experiences the most user load. Peak search load can be driven by different factors, like Monday morning scheduled reports, month-end accounting, or seasonal activity cycles.

Find your patterns and make sure you size and tune for those peak times.

Average CPU Usage

What it Indicates

The performance trend average CPU usage tells you how busy your deployment is as a whole. Looking at long time ranges here can tell you how much room you have to grow.

What to Look For

Narrow in on times with the highest CPU usage and make sure that your system does not often exceed 80% CPU on either indexers or search heads. Be sure that any places where your system is near or at 100% are short lived (seconds or less). Running above 80% puts your system close to resource exhaustion and can very quickly lead to failures. Running at 100% means your system is likely queuing requests, which can lead to even longer time spent at high CPU usage, leading to skipped searches, or server restarts.

If you do see high CPU usage, use the Resources Monitoring and Search Metrics sections to narrow in on which servers or what searches are using the memory.

Average Memory Usage

What it Indicates

The performance trend average Memory usage tells you how data-intensive your searches are. As your system ingests more data, the potential to use more memory increases.

What to Look For

A cold system will always consume memory rapidly as it fills caches. After several days running in a steady state, look for increases in memory.

If memory continues to increase, use the Search Metrics section to determine which searches are consuming more memory (or taking longer to run). A common cause is when searches use the “All Time” time range; as more data is ingested, these searches can scan over and return larger results.

System and Environment Data

System Environment

What it Indicates

The system environment tables list the versions of the installed Splunk software and add-ons.

What to Look For

Search Splunk support and communities for any issues with versions you are running, and upgrade them if necessary.

Data Inputs

What it Indicates

Data inputs gives you a view into where data is originating and where it is being stored. This can help you find any discrepancies in data routing.

What to Look For

Often data from a particular sourcetype will be directed to a specific index. You may notice that the distribution of events into different indexes is not what you are expecting. Large indexes can have a negative impact on performance and lead to incorrect search results.

Indexing

What it Indicates

The indexing section shows you activity in the various indexing queues. Ideally, these queues are relatively small, and similar in sizes.

What to Look For

If any of the 4 queues (Parsing, Aggregation, Typing, or Indexing) is much larger than the rest, it can suggest a bottleneck in that queue that can lead to reduced ingest capacity.

If the parsing queue is large, consider ingesting structured data types when possible. Also look for complex data transforms, especially ones involving regex.

If the aggregation queue is large, look for issues in the data that could lead to poor timestamp extraction.

If the typing queue is large, consider simplifying regex replacement logic or annotations.

If the indexing queue is large, look for bottlenecks at the destination (local or remote disk speed, or network transfer speeds). Increasing the number of hot buckets might also help when using storage with higher latency.

The final possibility is that the deployment is undersized and the queues are growing because the system is too small to handle the demand. If all attempts to reduce queues don't help, consider adding indexers.

Buckets

What it Indicates

This section shows the bucket counts and sizes in your system and can inform you of timestamp issues leading to poor search performance.

What to Look For

A large number of small buckets can indicate an issue. The fewer files that need to be opened for each search, the better. If warm bucket sizes are significantly below the bucket size settings in your configuration, this suggests that there is an issue parsing timestamps or that older events are being ingested along with newer ones in the same time span. Ensure timestamps are correct in your indexed data, and ingest event data with homogenous time spans.

When disk performance is an issue, large buckets can also be a potential performance problem. In some cases, storage will run better with more frequent writes of smaller amounts of data. If this is the case in your environment, consider lowering the maximum bucket size, to roll hot buckets to warm sooner. The SplunkOptimize process will handle consolidating these buckets to restore the benefits of fewer buckets.

Optimize your bucket size for the typical time ranges over which you search.

Learn more about buckets here:
https://docs.splunk.com/Documentation/Splunk/9.4.1/Indexer/Bucketsandclusters

Search Metrics: Metrics Overview

Search Concurrency

What it Indicates

This is quite possibly the most important metric to keep an eye on to ensure a stable system. Search concurrency is a search head (or search head cluster) statistic that shows how many searches are either being processed, or waiting to be processed. A spike in search concurrency above the system's capacity will quickly degrade performance and put the system in a state that can take a long time to recover from. Your system can process as many concurrency searches as CPU cores on a single indexer (if non-uniform, the smallest one); everything beyond that gets queued. Once search queuing starts, the recovery time back to stable depends on how many searches were queued during the spike and what the stable incoming search

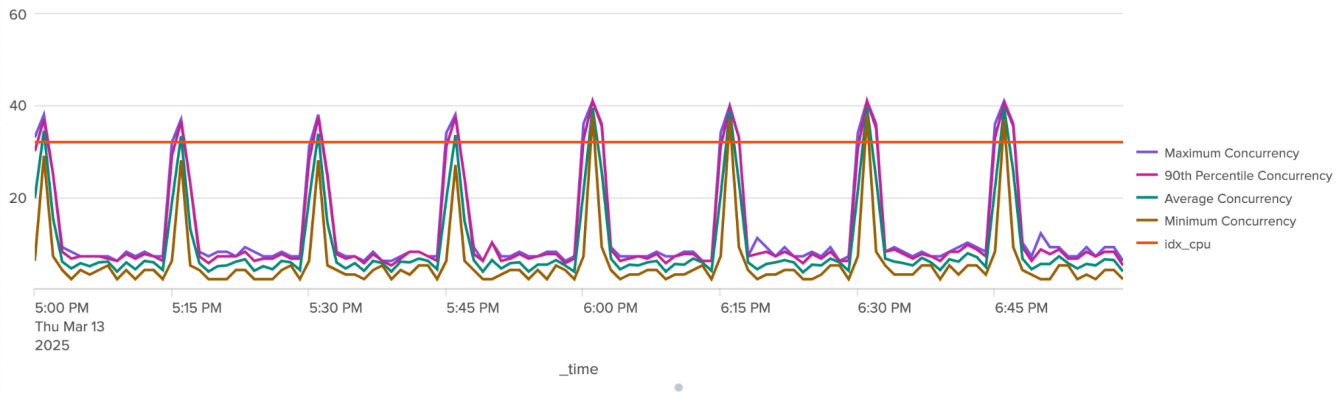
rate is. Think of it like going to grab a coffee just as a busload of tourists arrives in front of you: in just a few moments, a long line has formed and it will take time for that line to shrink again.

During this time of high concurrency, system resources become overburdened and search failures become increasingly more common. Other processes also take longer, leading to unexpected behaviors in other parts of the system (e.g., missing health check data, slower bundle replication).

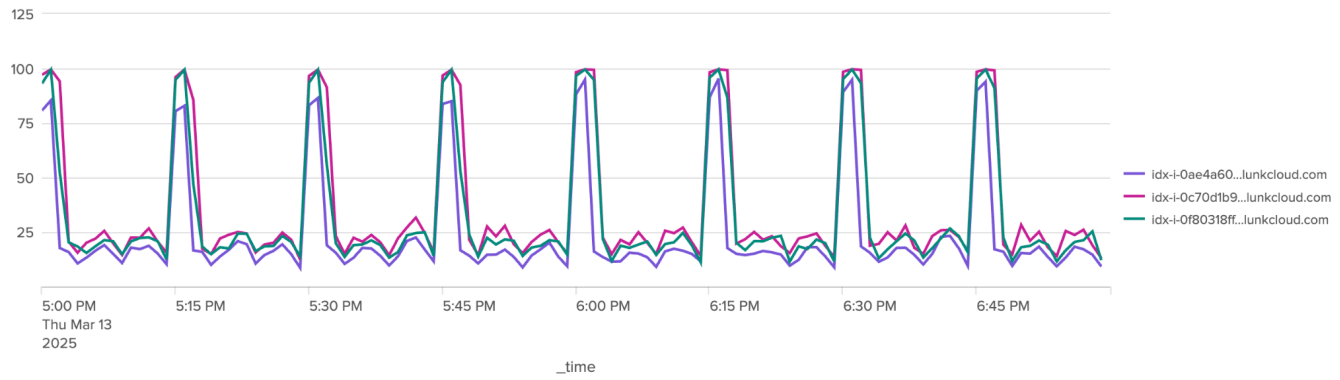
What to Look For

Any spike in concurrency above the CPU core count of a single indexer will lead to search queueing. Because of the method used to gather the search concurrency metrics, and the very fast nature of some searches, a small amount of search queuing can be tolerated. The magnitude and duration of the spike and the period between spikes can amplify the negative effects. It only takes a small amount of extra load to increase a spike from a few seconds to several minutes.

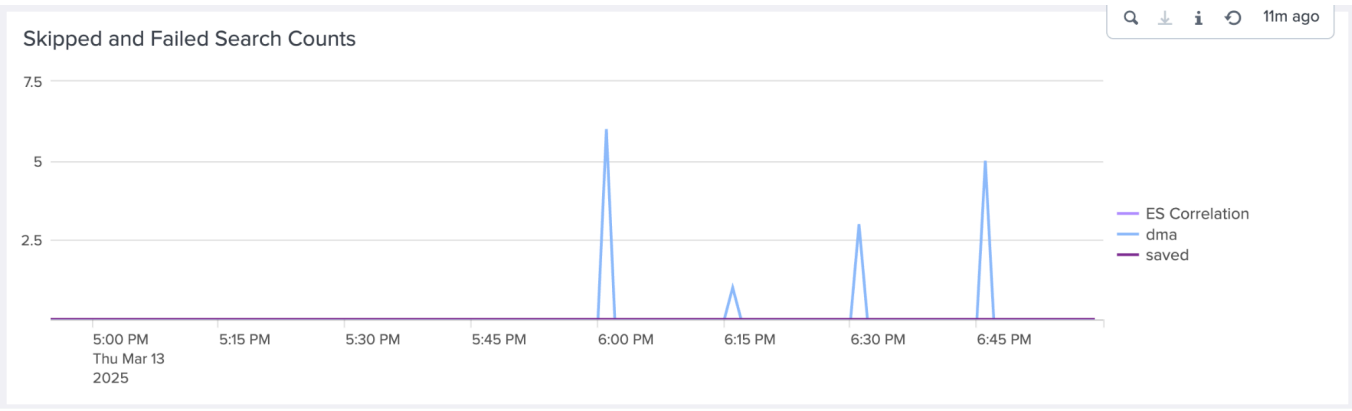
When using a lot of saved searches, or premium apps like Splunk Enterprise Security, it is common to see spikes in searches every 15 minutes, with the largest at the top of the hour. Spikes could also happen daily, or even yearly, so be sure to inspect an appropriate time range. To reduce these spikes, review your saved searches (including DMA and correlation searches) and where possible, change their CRON schedule so they start at a less busy time. You can also make use of schedule skewing and let Splunk spread the searches out over a defined time range.



The red line represents the number of CPUs on a single indexer: the maximum safe concurrency limit.



As the concurrent search counts begin to exceed the Indexer CPU count, CPUs are at capacity.



An increase of 16 searches per hour (less than 1%) can be the difference between none and several skipped searches.

The key to a stable splunk system is sizing the deployment for your maximum search concurrency. In the ideal case, search load can be smoothed to almost flat, and the average search concurrency and 90th percentile search concurrency will be close to each other.

Tip: Add an overlay to your search concurrency chart in PerfInsights showing your Single Indexer CPU count. Keep your search concurrency below that line for best performance.

Search Runtimes

What it Indicates

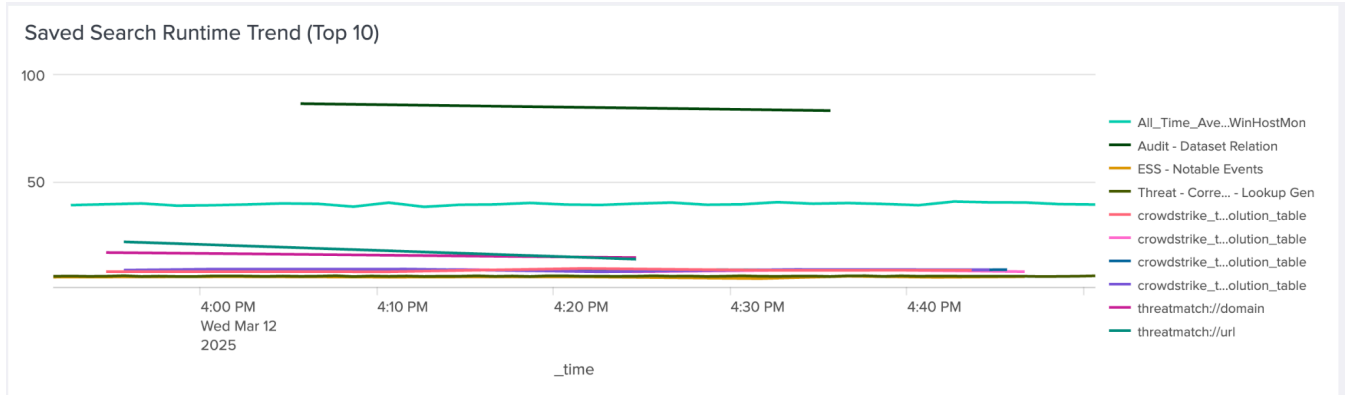
Search runtimes in this section give you a birds eye view of search performance.

What to Look For

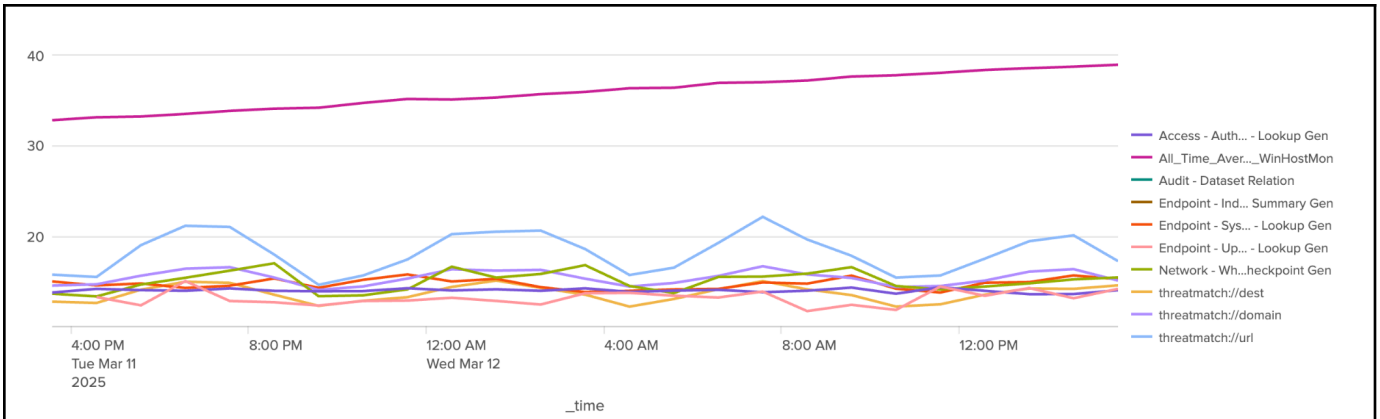
In general you want search run times to be as small as possible. Long ad hoc searches are the ones that users will likely notice the most. Ask yourself how long you'd be willing to wait for a search to return (e.g., maximum 60s), and try to keep your searches under that value. This might mean reducing the time period over which your searches run, improving the search SPL using more efficient search commands, or making use of base searches or job loading in dashboards.

It is normal for DMA and correlation searches to run longer, but they should still be less than one minute on average. Long running searches have a risk of running into the time allotted for the next search of its kind, leading to skipped searches. Disable any unnecessary and unused DMAs and correlation searches.

Watch out for searches that are getting longer over time. At first glance the runtime of a search over time might appear flat. Play with the timeline and chart scale to ensure this is the case. Searches that increase in runtime over time are likely scanning over all time. This can eventually lead to exhausted resources or search timeouts.



Over one hour, with the chart scaling to 100 seconds, all searches look stable.



Over 1 day, with the chart limited to 40 seconds, one search shows a potential problem.

Search Counts

What it Indicates

Search counts in this section give you a quick view on the trend on the currently selected time span compared with the previous span. This can tell you if recent changes to search load have taken effect.

What to Look For

Look for unexpected increases or decreases in search load from the previous period.

Skipped and Failed Searches

What it Indicates

Skipped and failed search charts give you a view into system stability. While there are a number of reasons for failures, both of these can indicate performance issues.

What to Look For

Check to see if skipped or failed searches correlate with excess search concurrency. If they do, then address that problem first. You may find this problem goes away.

If search concurrency is not an issue, then check search runtimes and failure causes for scheduled searches, especially for DMA and correlation searches. It could be that searches are skipping because they are taking too long and overlapping the next scheduled run time.

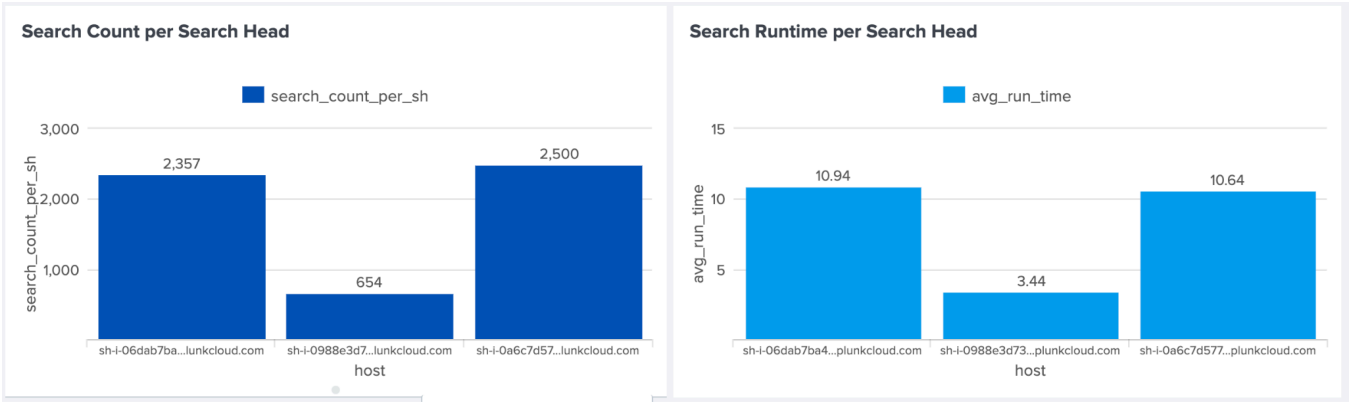
Search Counts and Runtimes per Search Head

What it Indicates

Breaking the search metrics out by search head allows you to see if any search head is experiencing a problem.
Depending on your configuration, in clustered search head environments, you may see one search head (the captain) with lower search volume than the others. That is normal. In any other case, a search head that is expected to be a peer of the others should be processing near the same amount of load.

What to Look For

Ideally, all things being equal, and ignoring the search head cluster captain, search load should be evenly distributed across all search heads.



It is normal for the search captain to differ in load, but the other search heads should be roughly equal.

If a search head is processing a very different amount of load than its peers, there may be configuration settings that are controlling that. Compare the server.conf values for the search heads to make sure you have the correct rules.

Search Metrics

These pages help you identify slow and inefficient searches, and can give some clues as to who the searches can be improved. Search efficiency is a topic too big for this document, but when you find searches that are taking a long time or using a lot of system resources, make use of other Splunk resources to improve them. These links are good starting points for that:

https://www.splunk.com/en_us/blog/tips-and-tricks/learn-spl-command-types-efficient-search-execution-order-and-how-to-investigate-them.html

https://lantern.splunk.com/Splunk_Platform/Product_Tips/Searching_and_Reporting/Optimizing_search

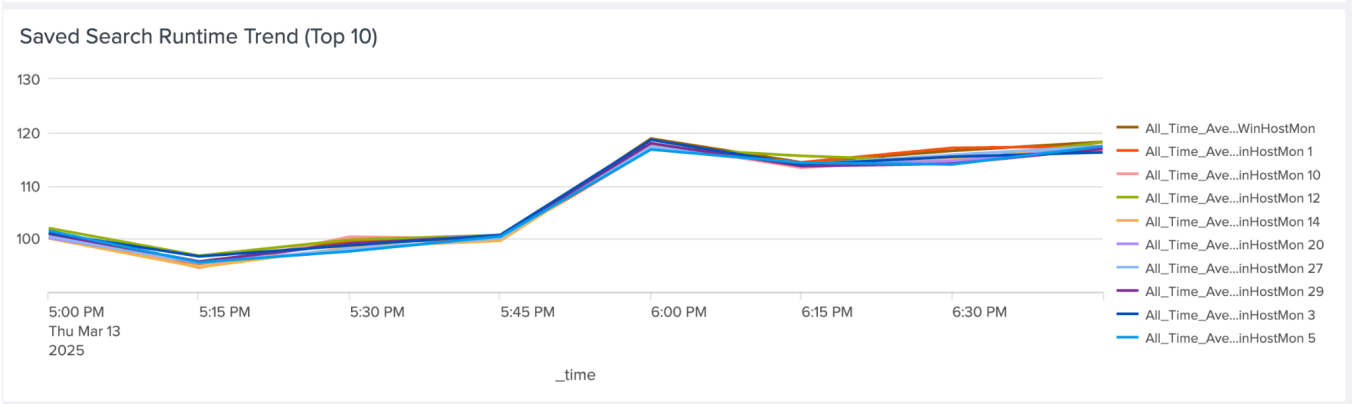
Search Runtime Trend (DMA. Correlation, Saved, and Ad Hoc Searches)

What it Indicates

This chart shows the top 10 most expensive (in terms of time) searches, and how they vary in runtimes over time. These are the searches that will have the most impact when looking to improve system efficiency.

What to Look For

Keep an eye out for any search times that are increasing over time. This could be an indication that resources are becoming exhausted, search concurrency is too high, or searches are searching over all time.



A small percent increase in search counts, at the wrong time, can lead to a large increase in runtimes.
In this example, the 1% increase in search load, led to a 15% increase in runtimes.

Search Performance

What it Indicates

The search performance table allows you to compare search time averages and extremes. The table also shows scan counts and results counts.

What to Look For

The closer the runtime extremes are to the averages, the better. Large gaps can indicate potential performance problems. Look for times where searches were running longer and compare them with other system behaviours at that time.

High scan counts, especially with low event counts, can indicate inefficiencies in your searches. When you see searches with high scan counts, attempt to lower those scan counts by adding more filters (in particular, indexed fields), and tightening time spans where possible.

If your event or result counts are unexpectedly low, you may be hitting configuration limits. Limits exist for saved searches, subsearches, and commands like join, or top. To configure limits, edit the advanced search properties, or create local *.conf files to override defaults.

Go here for more information on setting limits:

- Limits.conf
- Savedsearches.conf

Search Resource Usage (DMA. Correlation, Saved, and Ad Hoc Searches)

What it Indicates

Search CPU and Memory usage charts show which searches are using the most resources. Since all searches are competing for the same resources, anything you can do to reduce resource usage has a positive effect on the system as a whole.

What to Look For

For the most expensive searches, in terms of CPU and memory, look for efficiencies. For example, make sure indexed fields use the index equality operator (::), or avoid inefficient commands like “join”.

savedsearch_name	avg_idx_cpu	avg_sh_cpu	max_idx_memory	max_sh_memory
All_Time_Average_Mem_Perf_WinHostMon 33	56.91%	0.42%	360.84	236.42
Endpoint - Update Signature Reference - Lookup Gen	17.11%	0.13%	314.27	182.32
Audit - Sourcetype Readiness - Lookup Gen	13.40%	3.55%	443.28	205.17
crowdstrike_ta_build_mac_ip_resolution_table	4.86%	0.01%	557.75	193.01
crowdstrike_ta_build_userinfo_resolution_table	3.52%	0.01%	463.13	189.50
Access - Authentication Tracker - Lookup Gen	0.00%	6.05%	0.00	388.45

Searches consuming a lot of resources are good candidates to optimize.

For more information on optimizing searches, see the following topics:

- Using Summary Indexing
- Troubleshooting High Memory Use

Search Skip/Fail Rate/Details (DMA. Correlation, and Saved Searches)

What it Indicates

Skipped searches are never a good sign.

Ideally, the skipped search rate should be zero.

What to Look For

For any search type with a non-zero skip/fail rate, check the details to see the reason for the failure.

savedsearch_name	total	successes	last_failed_reason	concurrency_context	concurrency_limit	failed_count	failed_ratio
_ACCELERATE_DM_DA-ESS-ThreatIntelligence_Threat_Intelligence_ACCELERATE_	28	24	The maximum number of concurrent historical scheduled searches on this cluster has been reached	saved-search_cluster-wide	38	4	14.285714285714285
_ACCELERATE_DM_SA-ThreatIntelligence_Risk_ACCELERATE_	27	24	The maximum number of concurrent historical scheduled searches on this cluster has been reached	saved-search_cluster-wide	38	3	11.11111111111111
_ACCELERATE_DM_search_nike_extrahop_dm2_ACCELERATE_	27	24	The maximum number of concurrent historical scheduled searches on this cluster has been reached	saved-search_cluster-wide	38	3	11.11111111111111

Too much search concurrency of one type of search can cause other types to fail.

If you see that the maximum concurrent instances of that search has been exceeded, this usually indicates that the previous search is running too long and running into the scheduled time for the next search. You can make the search more efficient or, if your searches allow it, reduce the time range of the search or increase the time between scheduled searches.

If you see that the maximum concurrent searches for the cluster has been reached, you are likely experiencing other resource issues too. You can add admission rules or user limits on concurrency to avoid this error, but keep in mind that that will cause more ad hoc searches to fail. The better approach is to try to flatten any search spikes by spreading scheduled searches out more. You may also have to consider that your system is undersized for the amount of search load you are generating.

Long-Running Searches (Saved and Ad Hoc Searches)

What it Indicates

The long-running search table provides insight into which searches could be causing performance issues on the system. It is not uncommon to have searches that take several minutes to complete, like ones that cover large time ranges. Searches of that nature, while sometimes necessary, can have a detrimental effect on the system as a whole, as they consume resources and reduce capacity for other concurrent searches.

What to Look For

If you see a long-running search that is repeated frequently, attempt to optimize the SPL, or see if it makes sense to run it less frequently, or over a shorter time span.

Long-Running Saved Searches

Savedsearch Name	Search Runtime	Search Start	User
All_Time_Average_Mem_Perf_WinHostMon	118.81s	03/13/2025 18:02:31 +0000	admin
All_Time_Average_Mem_Perf_WinHostMon 3	118.57s	03/13/2025 18:02:50 +0000	admin
All_Time_Average_Mem_Perf_WinHostMon	118.20s	03/13/2025 18:47:50 +0000	admin
All_Time_Average_Mem_Perf_WinHostMon 12	118.15s	03/13/2025 18:48:01 +0000	admin

Optimize and reduce the frequency of long-running searches.

If you see a search that has been running for an unexpectedly long time (e.g., hours or days), check to make sure that your search limits and timeouts are being enforced. If you can, modify the SPL to avoid runaway searches in the future.

Frequently Run Searches (Ad Hoc Searches)

What it Indicates

Frequently run searches are a good place to optimize. Because these are ad hoc searches, you may not be able to change the SPL, but it is possible to change how users get to that data.

What to Look For

If you see expensive and frequently run ad hoc searches, consider creating a saved search with a macro to allow users to get at the data without having to run the search themselves.

Resource Monitoring

Search Head CPU

What it Indicates

Search head CPU is not often a problem area. Most of the heavy lifting in search is done by the indexers. The search heads are responsible for non-streaming and non-distributable streaming commands, but are often working on much smaller, filtered data sets.

What to Look For

If you do see high CPU usage on search heads, look to see if you can restructure your searches to filter out more data and move distributable streaming commands ahead of any non-streaming or non-distributable commands.

Search optimization is a large topic and can take time to master. In general, follow these guidelines:

- Decrease time ranges.
- Use only necessary indexes.
- Filter out as much data as possible before non-streaming or transforming operations.
- Filter on indexed fields and use the :: operator.
- Use the TERM() operator where possible for text searches.
- Use tstats instead of stats if you don't need the raw data.

These links can provide more information on this topic:

Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk

Writing better queries in Splunk Search Processing Language

Use CASE() and TERM() to match phrases - Splunk Documentation

Search Head Memory

What it Indicates

Search head memory grows as search queues fill, and with searches that return many results for non-streaming processing. Job retention times also have an effect on search head memory.

What to Look For

Search head memory growth can indicate a potential problem, and is often a symptom of high search concurrency. When searches are completed and their job lifetime has elapsed, memory is released. If memory is growing, then either search concurrency is increasing, or current searches are returning more results than usual.

If you see search head memory growth, try to restructure searches to do more filtering and aggregation on the indexers. You may also want to consider reducing the default job retention settings to allow memory to be freed sooner. As always, try to reduce the amount of concurrent search spikes.

The Key/Value Store is a possible exception to increasing memory concerns. For performance reasons, the KVStore is configured to use a large percentage of system memory. It is normal to see that grow for days to weeks after starting a system. Check

Search Head CPU and Memory By Process

What it Indicates

By looking at the per-process values for CPU and memory, it is possible to narrow down root causes. When investigating an issue, looking at the resource usage for each process can show you what types of searches are problematic.

What to Look For

In a stable system you should expect that CPU and memory are also stable. One exception would be when a system has recently started or undergone a major change; in those cases you may expect to see either a step (up or down) in the graphs, or a gradual increase in KV Store memory.

An increase in KV Store memory may be caused by caches continuing to fill. The database that backs the KV Store is allowed to use a large amount of system memory. It is not uncommon for this memory to grow for days or even weeks after a system restart. If memory grows beyond 20% of existing RAM, be sure to keep a closer eye on it. You may have some lookups that are growing quickly in some unbounded way.

Rolling Restarts

What it Indicates

Rolling restarts tell you when your system was restarted on purpose. System behaviors after a restart do not reflect the steady state, so knowing when a restart happened allows you to formulate more informed explanations for the given observations.

What to Look For

A single bar will appear on this chart when the search heads were restarted on purpose.

Running Search Heads

What it Indicates

At times system behavior may resemble behavior after a restart, but there was no restart. The running search heads chart looks for gaps in search heads logs that might indicate an unexpected restart of a search head.

What to Look For

The chart should remain steady showing the total count of search heads in the cluster. Any dip below that value might indicate a search head crash. It is possible, though rare, that this chart gives false positives showing a dip even when all servers were running. Use this chart to confirm suspicions of search head crashes.

Indexer CPU

What it Indicates

Searches are split over all indexers. If indexes are well balanced, each part of the search will complete in a similar amount of time. While processing a single search part, the indexer will dedicate one CPU fully to that task.

What to Look For

At times of high search concurrency, well balanced indexers will all be running at or near 100% CPU. Your goal is to minimize the time spent in this state to allow more searches to run, thus increasing search throughput. Minimizing time spent on the indexers is a matter of reducing the time range over which the search runs, and using as many indexed fields as possible for filtering. Ensure your searches do not query for excess data by searching beyond the range you need.

Spikes in CPU usage could indicate too many scheduled searches starting at the same time (for example, scheduled searches running at the top of the hour). If searches do not need to start at a particular time, try to spread their start times out.

Searches that run on the same schedule can cause CPU spikes, where a small amount of extra load can cause significant increases in search times and lower success rates.

If any indexer differs drastically in CPU usage, you may have unbalanced indexes. Make sure that your forwarders are sending data to all indexers instead of targeting a particular indexer (i.e., do not use IP addresses for indexers in your forwarders). You may also consider performing a data rebalance, but be aware that this is a lengthy process.

Indexer Memory

What it Indicates

Indexer memory is directly related to the combined ingest and search load.

What to Look For

If indexer memory is nearing its limits, be sure that you need (or may need in the future) all the data you are ingesting. Reduce the data range of your searches, and use more filters when possible. Also consider an instance type with more RAM.

Indexer CPU and Memory by Process

What it Indicates

By looking at the per-process values for CPU and memory, it is possible to narrow down root causes. When investigating an issue, looking at the resource usage for each process can show you what types of searches are problematic.

What to Look For

In a stable system you should expect that CPU and memory are also stable.

If CPU or memory are gradually increasing, your searches may be scanning more data over time. If you don't need it, try changing searches over All Time to a fixed time period instead.

User Limits

What it Indicates

The user limits page allows you to see values that are often quota restricted, either system-wide or per user. Sometimes users will experience failures and not know why, or possibly even be unaware of a failure at all. This can often be the result of a quota limit causing a silent failure behind the scenes.

What to Look For

Match user sage against user quotas to see if any quotas are being exceeded. Adjust quotas if necessary, or consider running saved searches under a different account and giving users access to the data via a dashboard or consolidated view.

Tip: Modify the chart SPL to include actual quotas as an overlay.

Splunk Features

Bundle Replication Size/Time

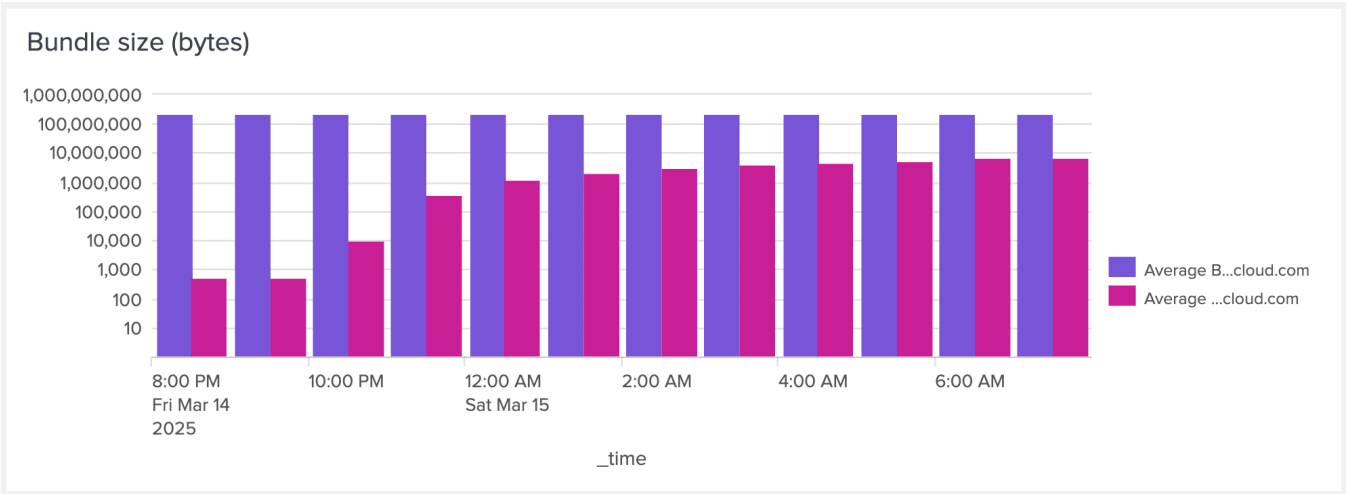
What it Indicates

The charts in the bundle replication section show you how knowledge objects are transferred between search peers. Replication times are usually fast (milliseconds), but an occasional replication taking hundreds of milliseconds is not uncommon. Changing lookups, modifying configurations, or installing apps will trigger longer replications.

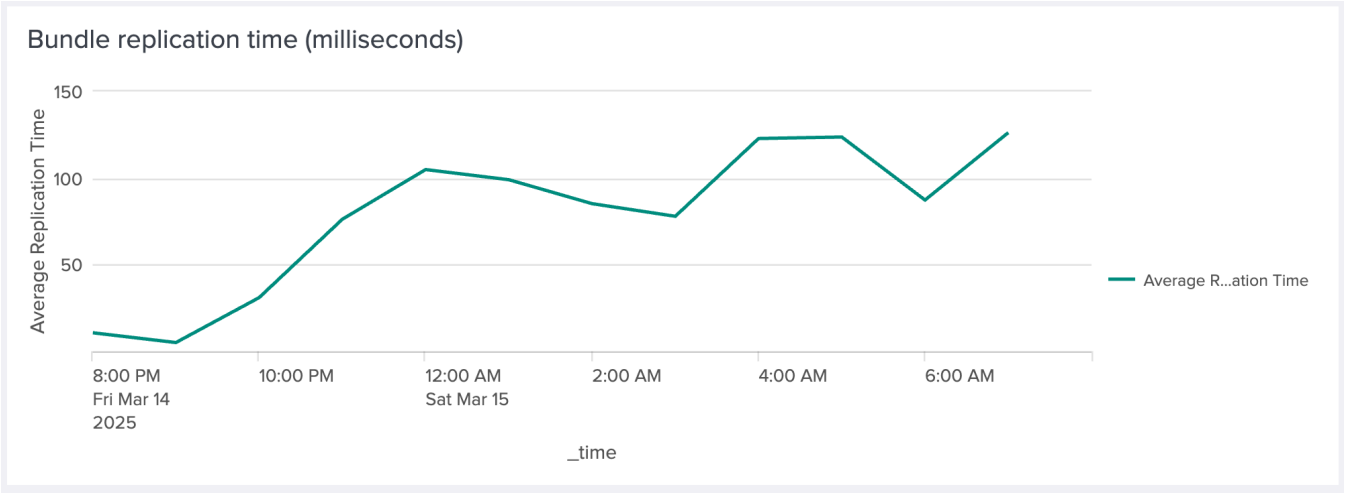
What to Look For

Stability is the key here.

If replication times or bundle sizes are increasing, check to make sure your lookups are not growing unexpectedly. While lookups can be dynamic, be careful not to allow unlimited growth. For example, if you add 20 bytes each to 10 fields in a lookup four times an hour, after a year, you will be replicating over 7MB every 15 minutes for this lookup alone. While 7MB can replicate quickly, it increases your overall replication times, and during peak times, missed replications can cause other usability issues.



Modifying lookups with tens of thousands of entries can cause bundle replication sizes to balloon from hundreds of bytes to megabytes instead. Try to keep lookups as small and as static as possible.



With increased bundle replication size, comes longer replication times, not to mention the possibility of increased data transfer costs.

Erratic bundle replication times could indicate network saturation, especially if there are many search heads involved (since each bundle needs to be replicated to every search peer).

Smart Store Performance

What it Indicates

If using SmartStore, these values tell you your data transfer rates and warm bucket eviction rates.

What to Look For

Slow data transfer speeds will affect performance. Work with your traffic engineers to ensure data transfer is configured optimally.

Slow eviction speeds might indicate disk resource contention on the local disk.

Cache Hit Rates

What it Indicates

A high cache hit ratio means that you aren't having to fetch data from the SmartStore very often.

What to Look For

If you see a large cache hit ratio, your performance is likely very good, but you may be putting strain on local resources. For tuning, see

https://docs.splunk.com/Documentation/Splunk/9.4.0/Indexer/ConfigureSmartStorecachemanager#Set_cache_retention_periods_based_on_data_recency

If you see a low cache hit ratio, you are likely evicting caches too frequently. For tuning, see

https://docs.splunk.com/Documentation/Splunk/9.4.0/Indexer/ConfigureSmartStorecachemanager#Set_the_eviction_policy

Bucket/Data Model Statistics

What it Indicates

Bucket and Data Model statistics tell you your upload, download, eviction, and removal rates over time.

What to Look For

Upload and removal charts are mostly about your ingest rate and retention settings.

Pay attention to download and eviction rates. If those are too high, you may be searching back over too large a date range; adjust your searches to look back over a smaller range, or increase your cache limits. If they are low, you may be using more local storage than you intended; decrease your cache limits.

Assets and Identities Statistics

What it Indicates

The assets and identities section shows you the size of those lookups.

What to Look For

Large lookups can cause a few performance issues including slow searches, and slow bundle replication. While support for these lookups exceeds hundreds of thousands of entries, if lookups start exceeding 50K entries, if it makes sense, consider separating the lookups into smaller ones.

Notable Events

What it Indicates

Generating and processing notable events triggers many actions beyond the scenes. An upward trend in notables can lead to performance issues.

What to Look For

Unless there is an active incident, correlation searches should be tuned such that notables remain steady over time. If there is an upward trend, or you suspect notables are too high, use the table to determine which correlation searches might need higher thresholds.

Environment Diagnose

Log Trends Overview, Distribution, and Top 100

What it Indicates

Log trends are a good indication of potential performance issues, even if the logged event itself has nothing to do with performance.

What to Look For

An upward trend in warning and error logging obviously indicates a problem. Events that cause errors and warnings can often come with retry logic that consumes system resources. Strive to eliminate all errors and as many warnings as possible. Zero is unrealistic, but the goal should be as few as possible. Focus on repeated errors that happen in close proximity as these are likely retries.

Conclusion

By giving you, the customer, the tools to diagnose performance issues in your own environments, you can reduce Time To Resolution, improve your users' satisfaction, lower reliance on Splunk Support, and address potential problems before they become problems.

Appendix A: Useful Links

Topic	Link
Bucket Size	indexes.conf - Splunk Documentation
Bundle Replication	Troubleshoot knowledge bundle replication - Splunk Documentation
Configuration Files	List of configuration files - Splunk Documentation
Efficient SPL	Learn SPL Command Types: Efficient Search Execution Order and How to Investigate Them | Splunk
	Writing better queries in Splunk Search Processing Language
	Use CASE() and TERM() to match phrases - Splunk Documentation
Indexer Health	Restart the entire indexer cluster or a single peer node - Splunk Documentation
Report Scheduling	Schedule reports - Splunk Documentation
	Use cron expressions for alert scheduling - Splunk Documentation
Search Head Health	Use the CLI to view information about a search head cluster - Splunk Documentation
Search Windowing and Skewing	Schedule Windows vs. Skewing | Splunk
User Limits	Authorize.conf