Quis Custodiet Ipsos Custodes? (Who Watches The Watchmen?)

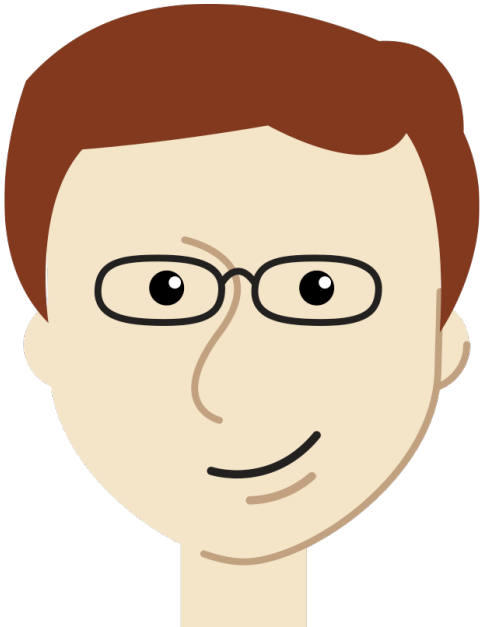
Or, How Do You Know When Splunk Stops Searching?

Tom Kopchak
Hurricane Labs

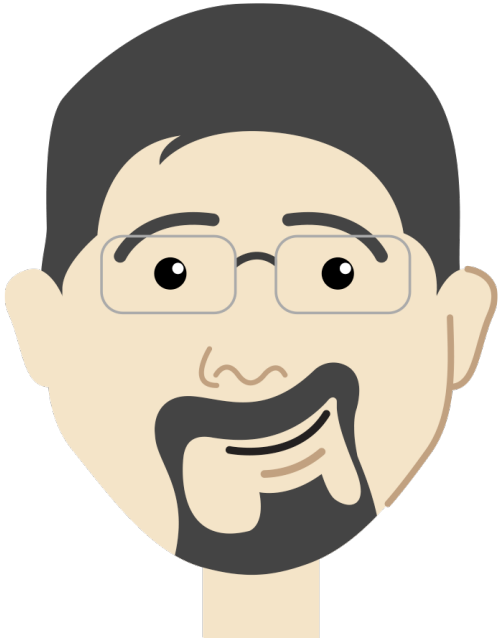
Tim Baldwin
Hurricane Labs

About Us

Tom
tom@hurricanelabs.com

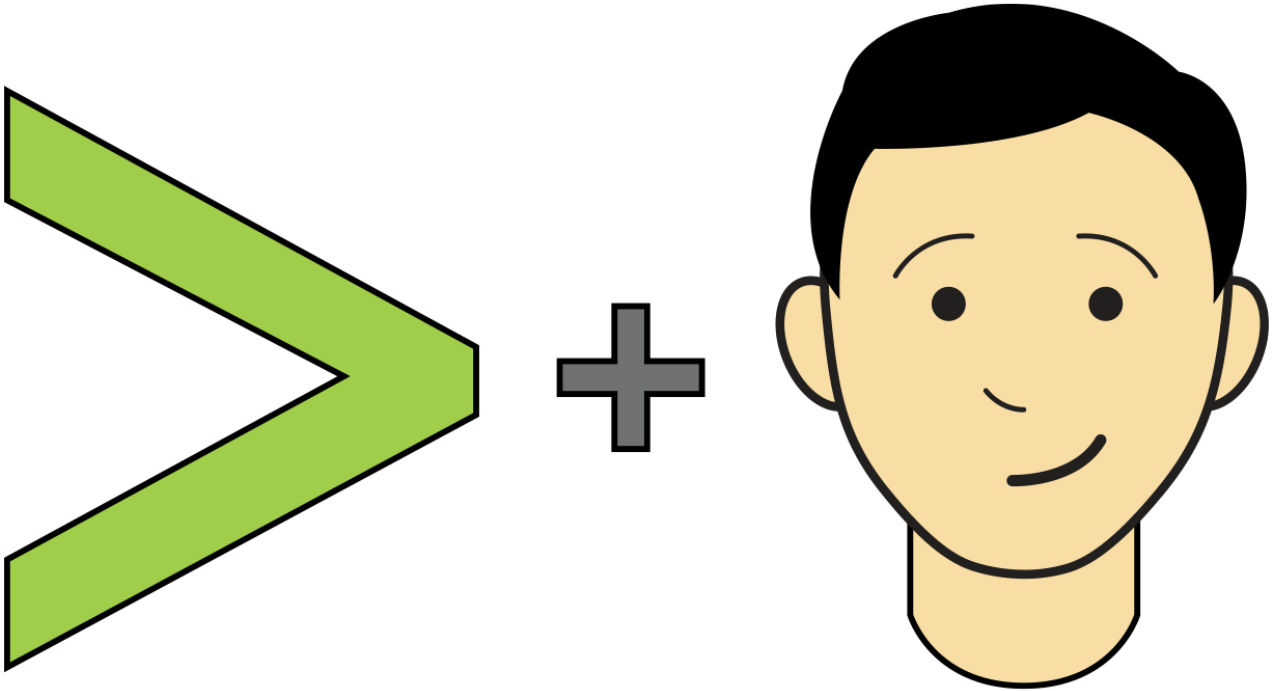


Tim
tim@hurricanelabs.com



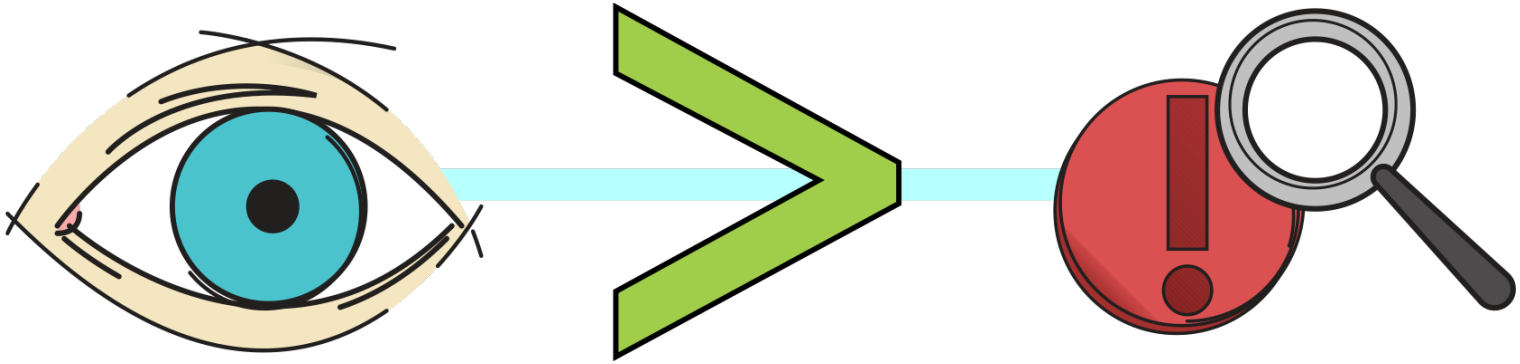
About Hurricane Labs

Splunk And You



What Will Be Covered

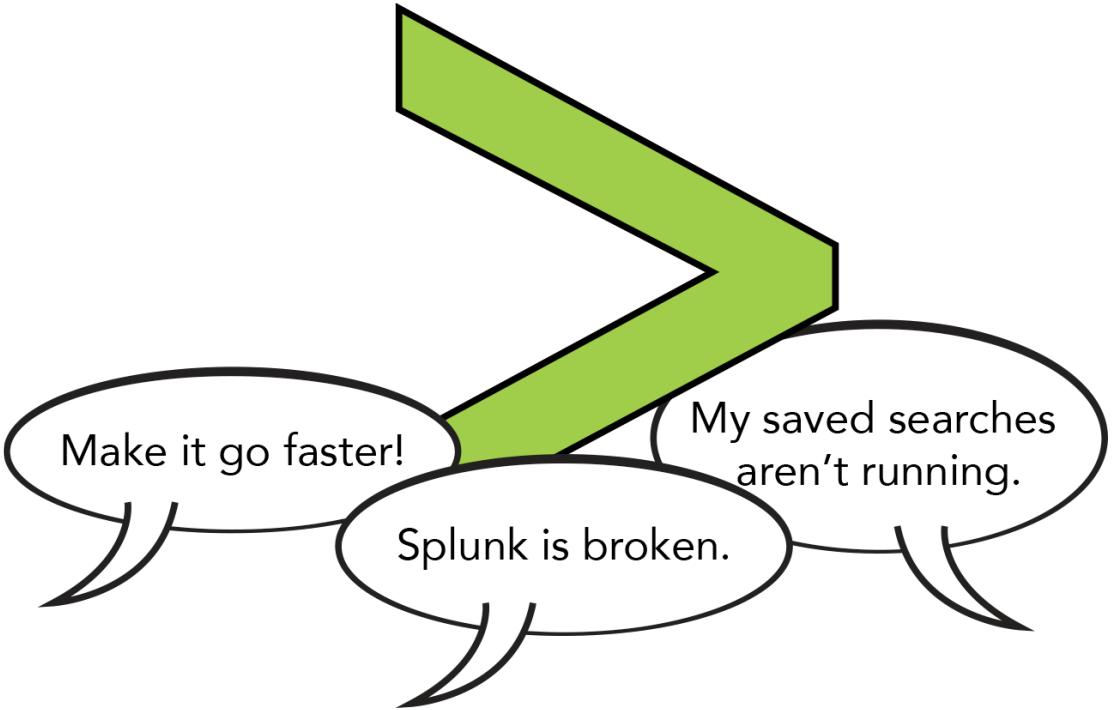
- Best practices around Splunk monitoring and alerting
- Types of monitoring available
- The difference between monitoring and *good* monitoring



What Will NOT Be Covered

- This is NOT an end-to-end walkthrough
- We will NOT tell you which monitoring and/or alerting platform to use
- We will NOT show the specific configuration file changes that will be needed

How Many Of You Have Heard This Before?



Wouldn’t it be awesome if users never experienced a Splunk issue?

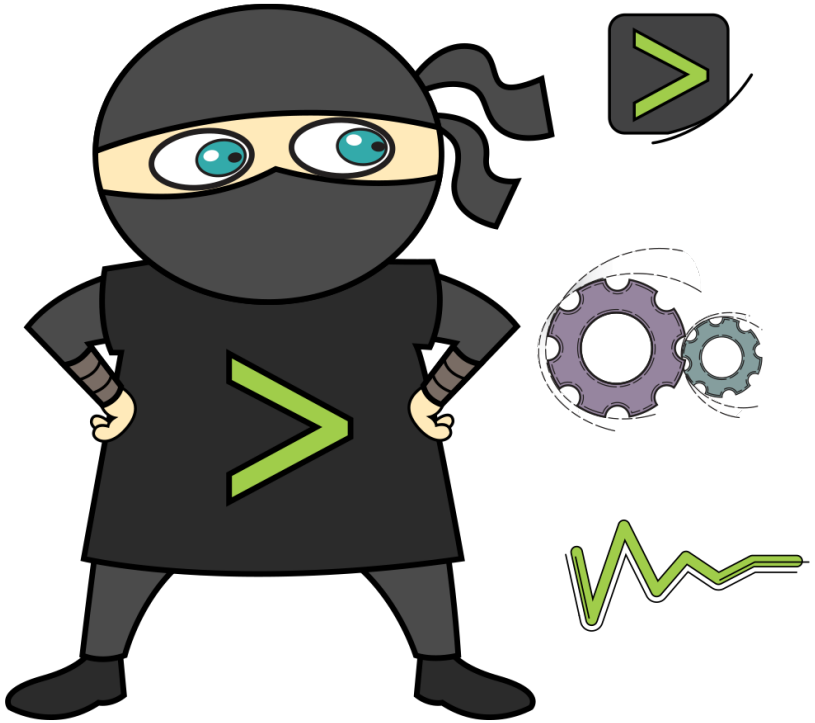
How Does Sam Do It?

- Who is Sam? Sam is a (gender neutral) Splunk ninja
- She used to be reactive
- Now, he is proactive

Sam's Story

- Sam used to just assume that all problems could be solved by users writing better searches
- As she gained more experience, he learned that is not always the case
- She now monitors Splunk so that he can solve problems more quickly, often before his users even notice the issue

What Does Sam Monitor?



Splunk Data

- Splunk Saved Searches/Splunk API

Splunk-related Processes/Services

- Splunk API/Operating System agent

Splunk-related OS operations and/or settings

- SNMP/OS Agent

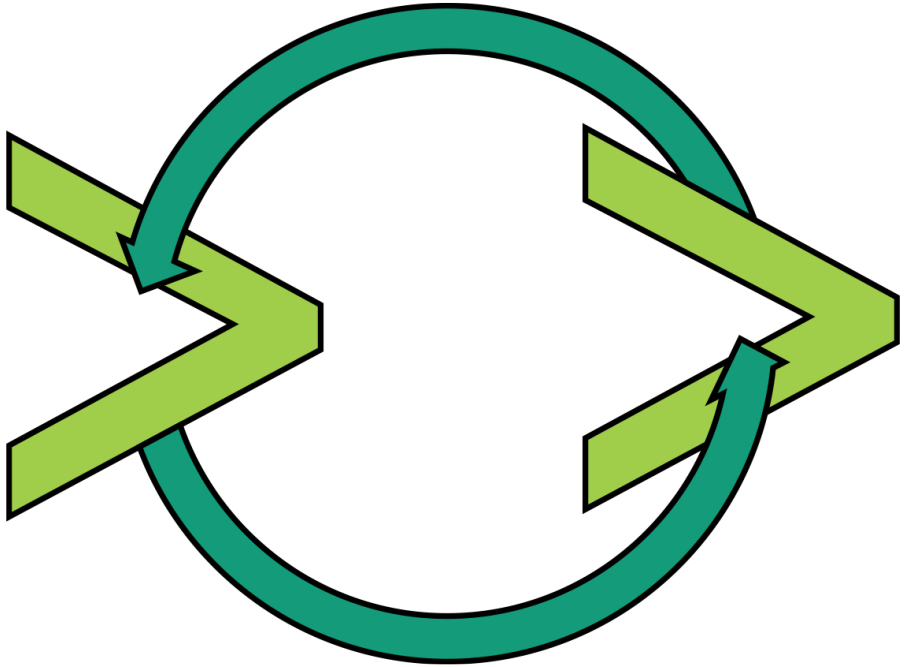
Normal OS operations and/or settings

- SNMP/OS Agent

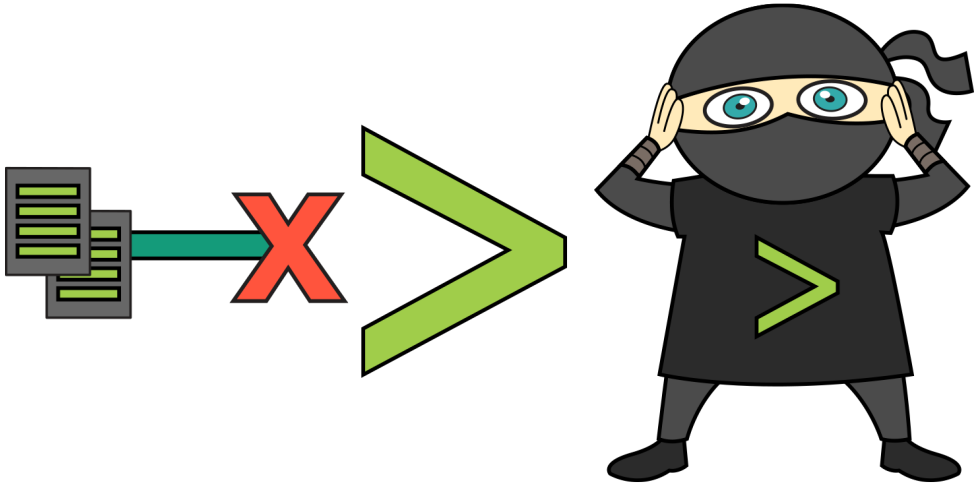
Monitor Splunk Via Splunk

Problem: Sam's user called up to complain about not being able to find firewall data

Sam spent two days trying to "help" the user write a better search to "find" the data



Monitor Splunk Via Splunk



Finally found that the data stopped flowing into Splunk

Fixing the issue was easy once it was discovered

Sam realized that (s)he could have prevented a ticket if he/she was monitoring for data that stopped coming in

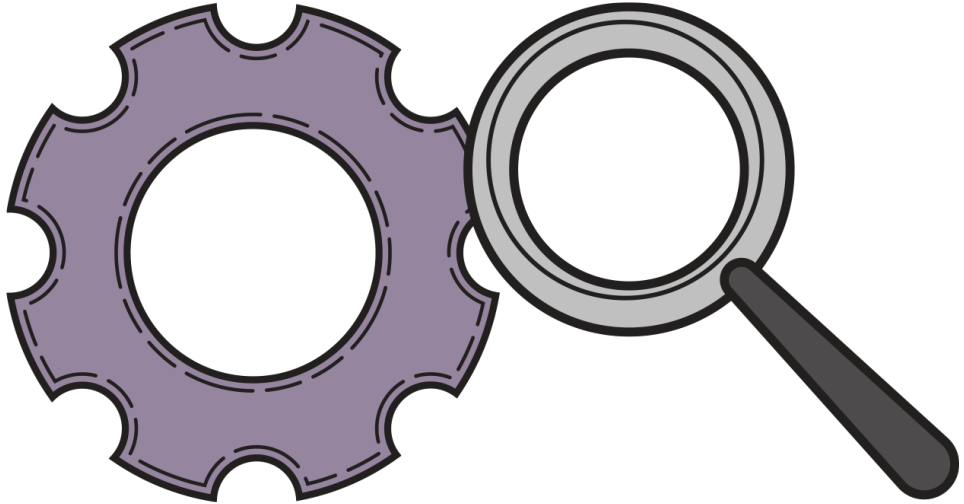
Monitor Splunk Via Splunk

Splunk Searches

- Broken Sources Sanity Check
- index=_introspection
- | REST

DMC

- Splunk Distributed Management Console
- Processor and Memory
- Licenses - Expiration and Quota Usage
- Missing forwarders
- Disk Usage
- Processing Queues
- Search Peers



Finding When Data Stops Flowing

Broken Sources Sanity Check

- Runs a search using "| metadata" to pull last time that a host sent data
- Is "tunable" using a lookup table
- Available on splunkbase: https://splunkbase.splunk.com/app/3247/

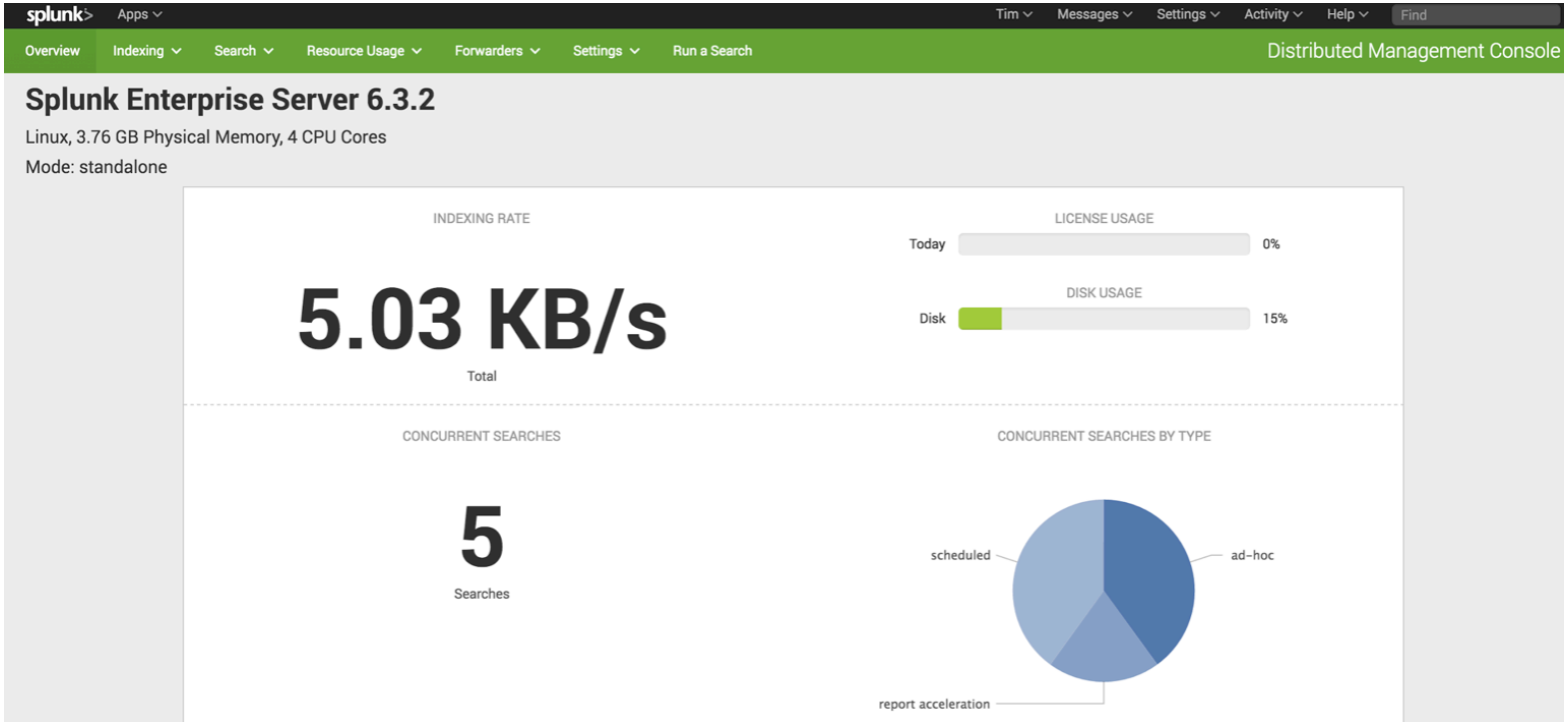
index=_introspection

- Resource usage on a per-search or system-wide basis
- Disk utilization
- Example in the Appendix

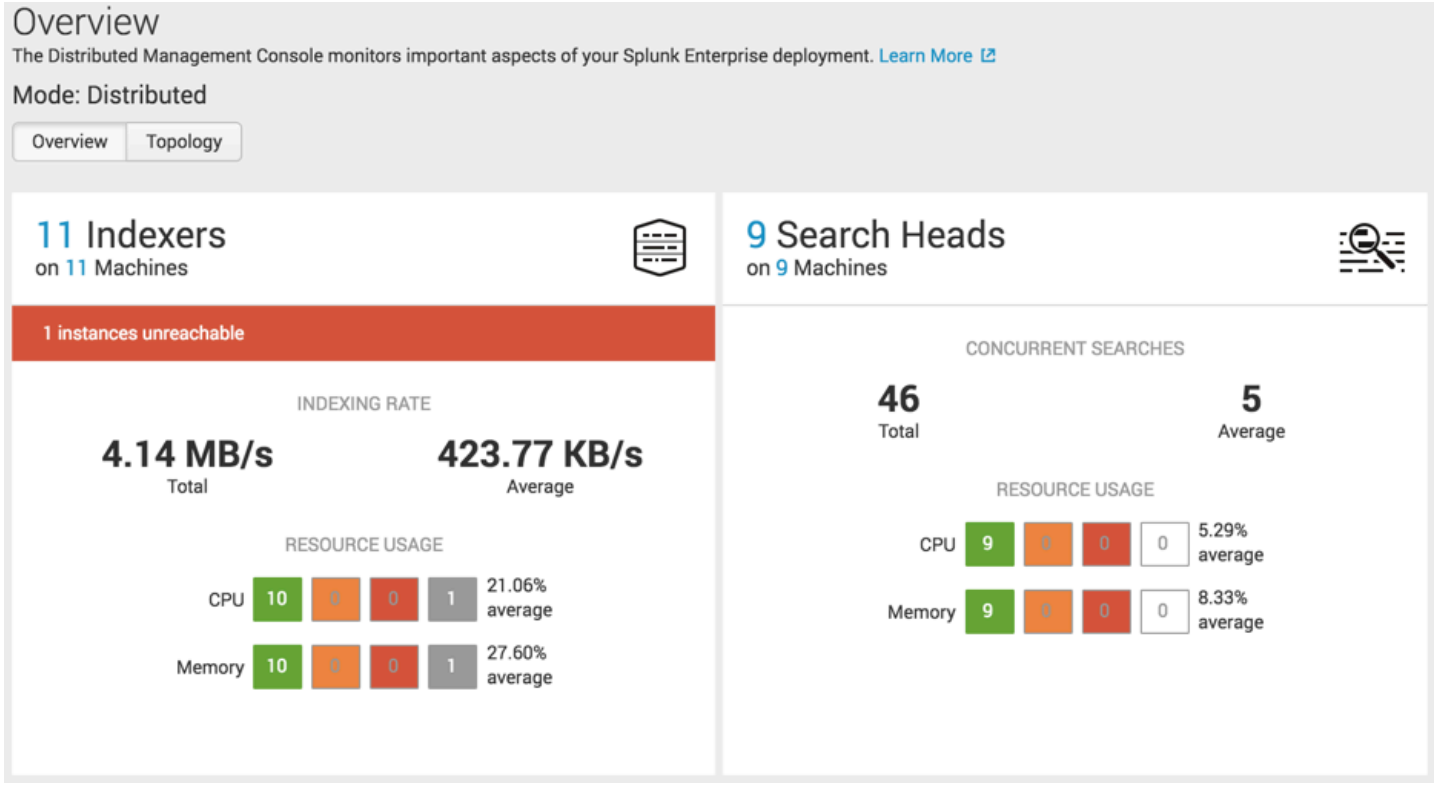
| REST

- Pull information about Splunk system health, license utilization, etc.
- Example in the Appendix

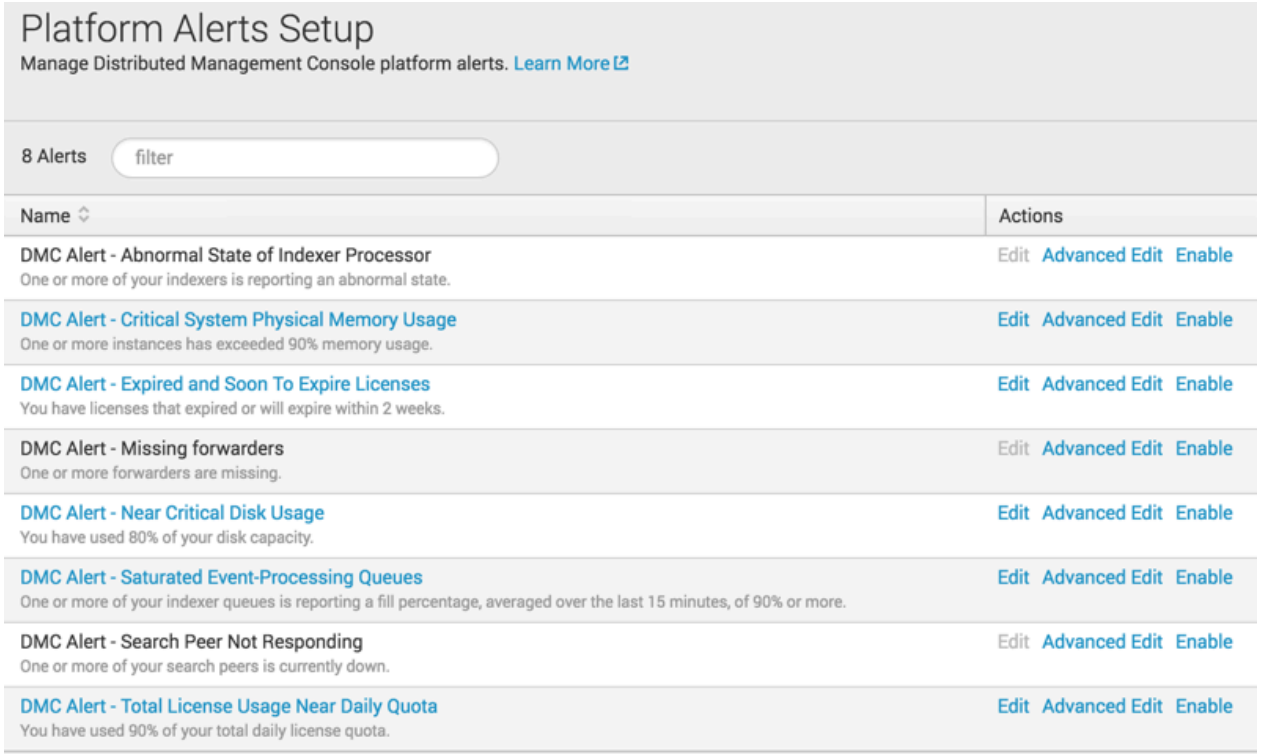
Splunk Distributed Management Console (DMC)



DMC Distributed Mode



DMC Alerts Setup

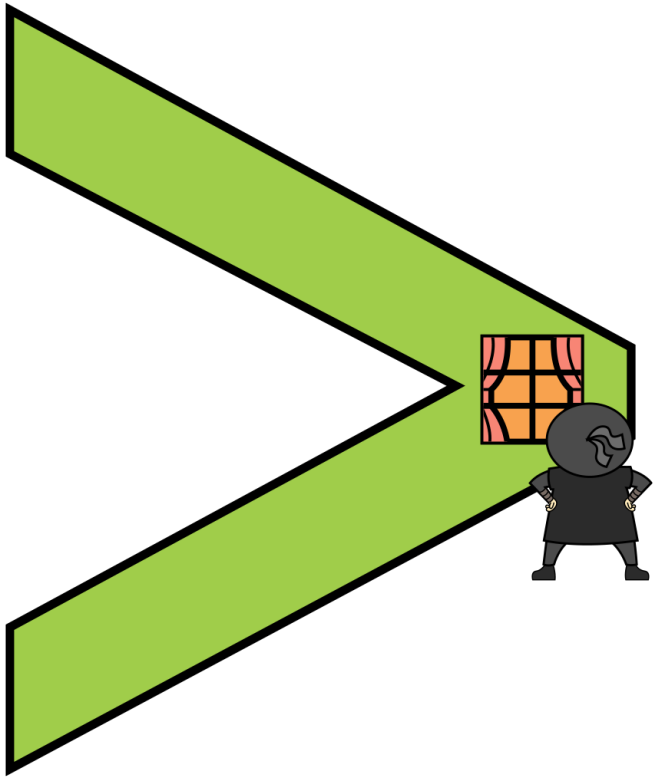


Monitor Splunk Outside Of Splunk

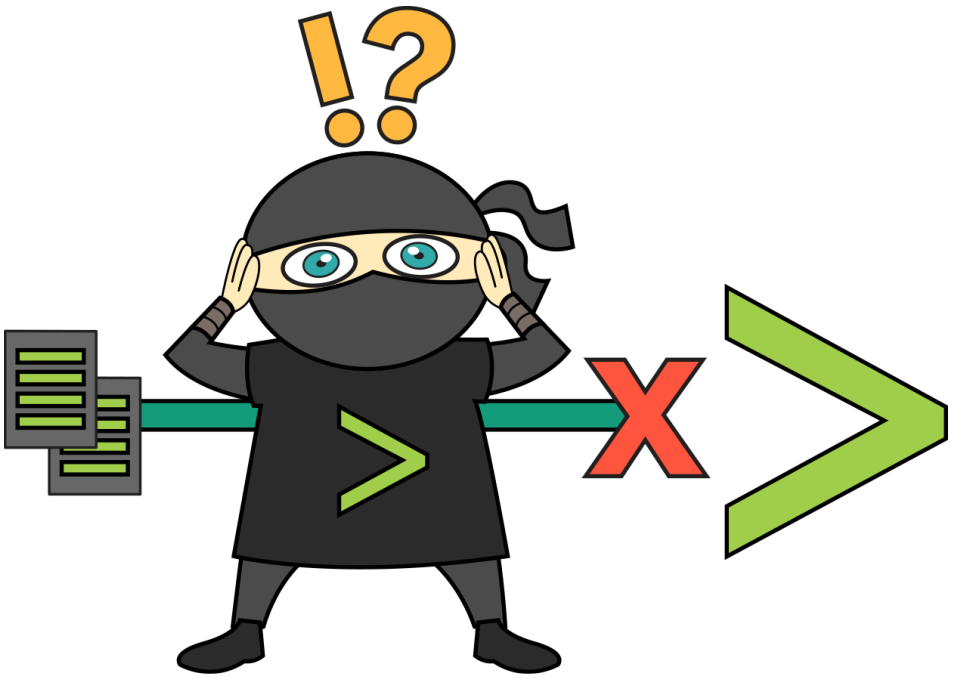
Problem: Sam's user called up to complain about not being able to find firewall data…. Again

Sam didn't get an alert from the Broken Sources Sanity Check

Sam reverted to the default assumption that the user was running poor searches



Monitor Splunk Outside Of Splunk



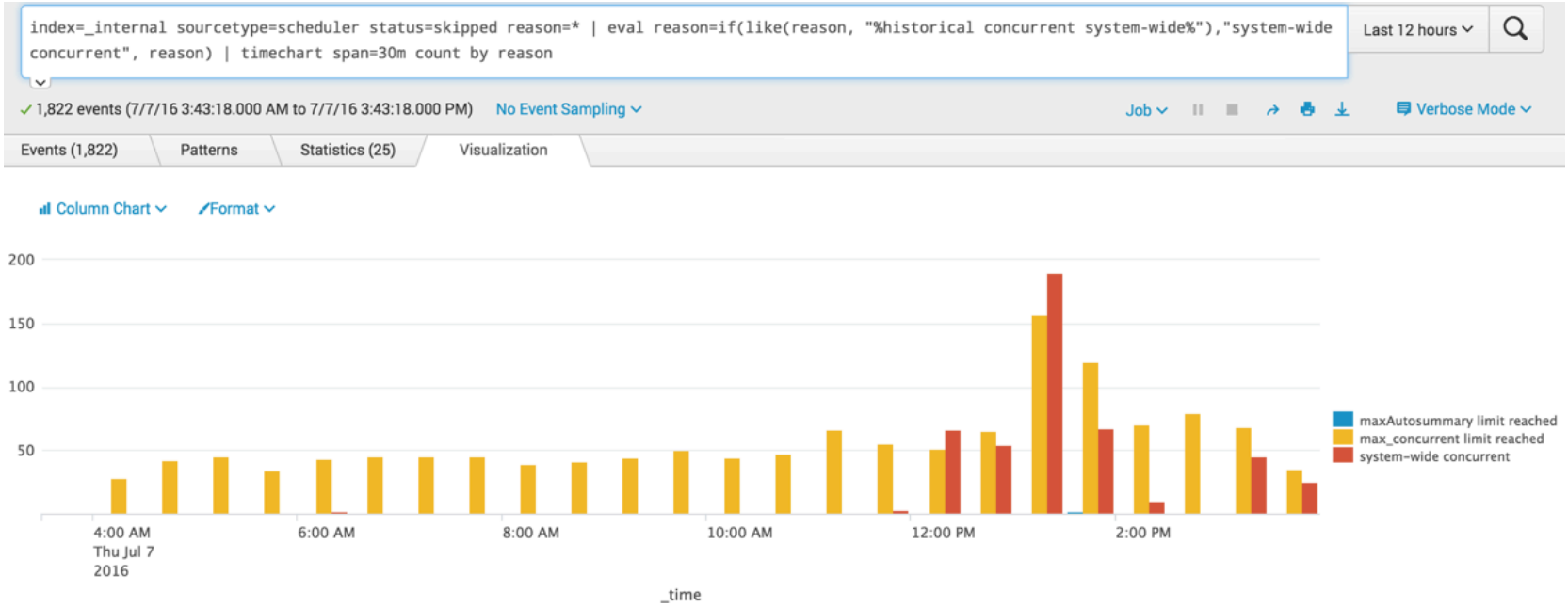
Sam finally found that the data stopped flowing into Splunk

- Same problem as before

But wait - didn't we have monitoring for this?!?

What's Going On?

The Broken Sources Sanity Check didn't alert because the search was getting skipped



Monitor Splunk Outside Of Splunk

Direct REST API

- Use REST API endpoints to show and alert on certain information
  - Listens on port :8089 by default
  - Could replace some of the "| REST" searches to reduce the search concurrency
- Splunk Messages
- Licensing information
- Deployment Client status
  - Check that specific clients are checking in
- Indexer Cluster Search factor/Replication Factor
- Indexer Cluster Node status
- Many others
  - http://docs.splunk.com/Documentation/Splunk/latest/RESTREF/RESTprolog

So, We Should Be Good, Right?

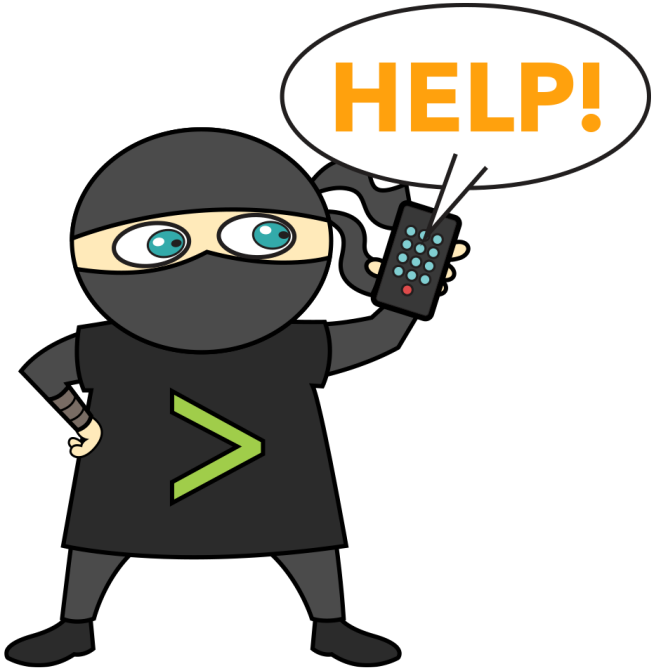
Monitoring Splunk using Splunk Searches

Monitoring Splunk outside of Splunk (REST API)

Monitor Splunk-related OS Settings

- Problem: Sam's user called up to complain about not being able to find firewall data…. Again?!
- She did not get an alert from the Splunk Searches
- But, he did get an alert from the Splunk REST API checks:

Splunk GUI message: "Cannot write data to index path "/mnt/splunk_warm/firewall/db" because you are low on disk space on partition "/mnt/splunk_warm". Indexing has been paused. Will resume when free disk space rises above 5000MB."



Monitoring Vs. Good Monitoring

There **is** a difference

Alerts should be:

- Relevant
- Timely
- Actionable

Monitor Splunk-related OS Settings

Found that the splunk_warm partition
(/mnt/splunk_warm) has very little disk space available

But wait - didn't the REST API check notice this ?!?

- Yes, it did - but not timely
- Sam needs to know BEFORE Splunk stops working

Sam realized that (s)he could have prevented a ticket if he/she was monitoring for Splunk-related partitions and processes

Monitor Splunk-related OS Operations Settings

Disk Usage

- Could replace the index="_introspection" searches to reduce search concurrency
- $SPLUNK_HOME disk space
- Splunk hot/warm/cold disk space

Processes

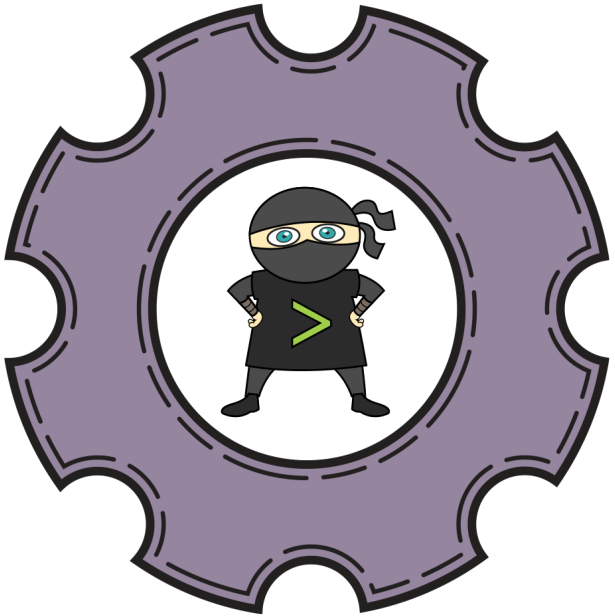
- Splunk processes
- Syslog-ng processes
- API processes (Java bridge, for example)

Listening Ports

- 443 or 8000
- 8089

HTTPS checks

- Cert expiration

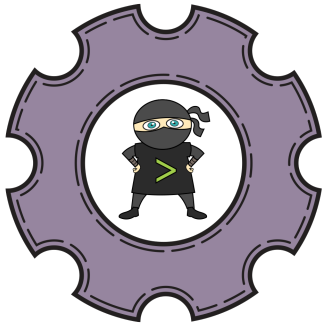


So, We Should Be Good, Right?

Monitoring Splunk using Splunk Searches

Monitoring Splunk outside of Splunk (REST API)

Monitor Splunk-related OS operations settings



General (Non-Splunk) Operating System Things

Problem: Sam's user called up to complain about not being able to find firewall data…. Again!!!!

She did not get an alert from the Splunk Searches

He did not get an alert from the Splunk REST API checks

There was no alert from any of the Splunk disk or process monitoring checks

General (Non-Splunk) Operating System Things

- Finally found that the root partition (/ or C:\ ) of the operating system filled up
- But wait - didn't Sam have disk space monitoring?
- Sam realized that (s)he could have prevented a ticket if he/she was monitoring for issues on the Operating System

General (Non-Splunk) Operating System Things

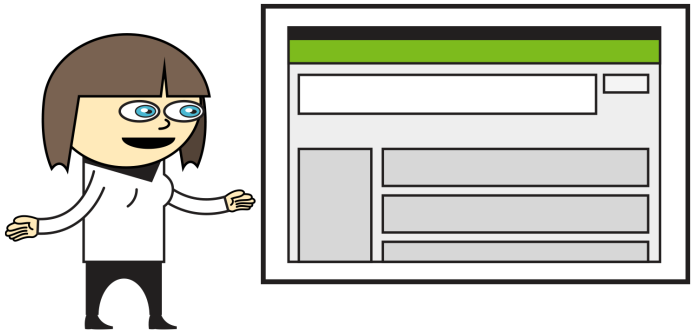
- Disk Space and mount points
- Disk/RAID health
- SSHD Process
- Port 22 listening
- Server uptime
  - Alert if system was unexpectedly rebooted
  - Alert if system hasn't been getting rebooted during patching cycle
- Memory usage
- CPU usage
- Load Average
- System Time - Important that NTP is working for Splunk
- Network interface(s)

General (Non-Splunk) Operating System Things

- Sam's CIO was giving live demo to the CEO to request funding for a larger Splunk license

- Sam was alerted that firewall logs stopped coming in and fixed the issue within minutes

- CEO was so impressed with the cool stuff they were doing with Splunk that she gave them funding to double their Splunk license!



Sam No Longer Has Users Calling With Issues

- Sam is now alerted before the users notice
- Sam is proactive
- We should all be more like Sam

What's Next? Active Response

- Monitoring system can trigger proactive action
- This is the future state of monitoring
- Examples:
  - Restart SSHD if cannot connect
  - Run ntpdate if time is not synced with NTP
  - When disk space gets low, automatically open a ticket with the storage team
  - Reauthenticate API when the API key expires
  - Restart Splunk if no splunkd processes are running

Wrap Up And Questions

Appendix – Slide #15

Slide # 15

- **Broken Hosts App for Splunk**:
- https://splunkbase.splunk.com/app/3247/

Appendix – Slide #15 (Continued)

Slide #15:

index=_introspection search example:

- index=_introspection sourcetype=splunk_resource_usage component=Hostwide | timechart Median(data.cpu_system_pct) AS "System CPU" Median(data.cpu_user_pct) AS "User CPU"

| REST search example:

- | rest /services/licenser/licenses

Additional REST endpoint information:

- http://docs.splunk.com/Documentation/Splunk/latest/RESTREF/RESTprolog

Appendix – Slide #21

Slide # 21

- **Simple**: index=_internal sourcetype=scheduler status=skipped
- **Advanced**: index=_internal sourcetype=scheduler status=skipped | eval reason=if(like(reason, "%historical concurrent system-wide%"),"system-wide concurrent", reason) | timechart span=30m count by reason

Appendix – Slide #25

Slide #25

- **Relevant**
  - Reduce false negatives and reduce false positives
  - Never alert when it's not a problem and always alert when it is a problem
- **Timely**
  - Be as proactive as possible
  - Not too early but not too late
- **Actionable**
  - If there's nothing that can be done, then it should not be an alert
  - We may want to know about trending issues if they persist

Appendix – How We Monitor

Checked with SNMP:

SNMP Time
SNMP Environment Status
Disk/Partition Free Space
Memory/Swap Usage
Load Average
Uptime
Interface Status

Checked with HTTPS connection:

HTTP/HTTPS availability
CVE-2009-3555 (TLS Renegotiation) Vuln
CVE-2011-3389 (BEAST) Vuln
CVE-2014-0160 (Heartbleed) Vuln
CVE-2014-3566 (POODLE) Vuln
SSL Certificate Expiration

Checked with Splunk Search:

Broken Sources Sanity Check
DMC Alerts
Notable Event Outage

Checked with REST API:

License Master Connection
Splunk Messages
Concurrent Searches
Search Peer Connection
Deployment Client Status
License Usage
Cluster Replication Factor Status
Cluster Search Factor Status
Cluster Peer Status
Cluster Maintenance Mode

Checked with port scan:

Splunkd TCP Port (8089) available
SplunkWeb Port

Checked with host agent:

SSH Port
Splunkd Process
Syslog Daemon

THANK YOU

splunk>