Splunk Boot Camp

Aggregate, analyze, and get answers from your data

v2.0-p

Meet Your Instructor

- Christian Ludwig
- Fractional CTO / Educator
- Rocket Nerd

cludwig@instantbrains.com
@papaludwig
linkedin.com/chris_ludwig

Introductions

Now it is your turn! Help me to understand who you are, your needs, your expectations, your background, and something interesting.

1. What is your Name?
2. What team are you on?
3. What is your role?
4. What is your experience level with Splunk or similar tools?
5. What is your expectation for this class?
6. What is something fun and/or interesting about yourself?

Questions? Please ask!

I am here to help you with your journey into and through Splunk. Although I may not always have an answer, the desire is to assist you, directed at your needs as much as possible. Ask questions as needed. **There are no bad questions!** It is about you and your needs.

Why do you think it is that way?

You are going too fast, can we revisit that?

How does that work?

Can we expand on that topic?

Do you have a recommendation for this?

What have you seen in the industry regarding this?

Are there best practices?

Is this what you mean by that?

Here is how we are trying to solve that problem. What do you think?

Where can we find more info?

Introduction to Splunk

Part #1

Introduction to Splunk

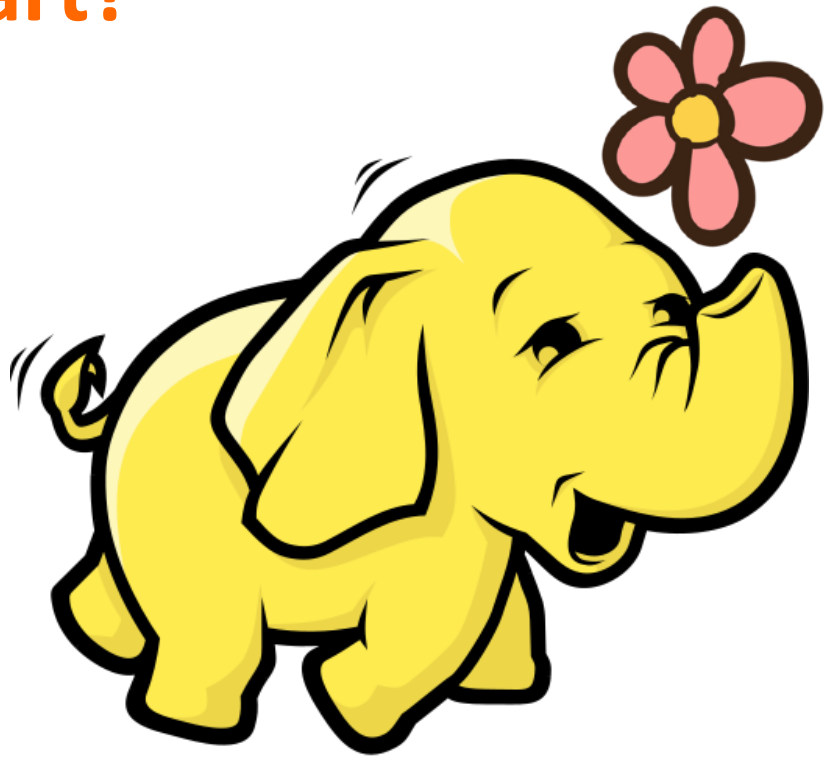
- What's Splunk?
- What Sets Splunk Apart?
- Users, Roles, and Capabilities
- Splunk Varieties
- Exercise: Lab Environment + Quick Tour

What Is Splunk?

- Spelunking - cave exploration
- First Released - 2003/2004 (!)
- Parse, Ingest, Search, Manipulate, Analyze, and Visualize Disparate Big Data in Real Time
- Robust Use/Administration through Web, CLI, API
- Management / Extension via Apps/Add-ons

What Sets Splunk Apart?

- Schema-less Storage
  - Term-based search, Bloom filters, time-series indexes, raw parsed data
  - “The Future is Federation”
- Search-time Schema as needed
- Supports massive parallelization throughout
  - Hadoop under the hood
  - Meaningful clustering into the thousands of scale units



Things To Do With Splunk

- Security Information and Event Management (SIEM)
- Application Log Management
- Artificial Intelligence for IT Ops (AIOps)
- Generalized Machine Learning
- Compliance
- Generalized Map/Reduce Work
- Data Viz
- Process Control Visualization / Management
- Bee Keeping
- Beer Brewing
- Surf Predictor

Splunk: What We Cover / What We Don't

- **Forwarding/Parsing/Indexing**
- **Search**
- **Reports**
- **Dashboards**
- **Alerts**
- **Data Model / Data Tables / Pivot**
- **Machine Learning Toolkit / Data Science & Deep Learning**
- **Some Cloud-Specific Features**

Users / Roles / Capabilities

- Single sign-on translates to Splunk User
- Users may be assigned to Groups
- Groups/Users are assigned Roles
- Roles have Capabilities
- Out-of-the-Box Roles:
  - Admin - Full system control, minus “can_delete”
  - Power - Create most constructs
  - User - Create some constructs / Use most constructs
- Fully customizable by Admins

Splunk Varieties - Enterprise / Cloud

- Splunk Security
  - Splunk Enterprise Security
  - Splunk Asset & Risk Intelligence
  - Splunk SOAR
  - Splunk Attack Analyzer
  - Splunk User Behavior Analytics

- Splunk Observability
  - Splunk Observability Cloud
  - Splunk IT Service Intelligence
  - Splunk AppDynamics
    - Still fully supported

- Pricing Options:
  - Ingest
  - Workload
  - Entity
  - Activity
  - splunk.com/pricing

Splunk Free - Because People Ask :)

- < 500 MB/day Index Volume
- Single instance deployment
- No authentication methods (no login)
- No HTTP/TCP Forwarding
- No Scheduling / Alerts

Splunk Cloud Platform

- Advertised as “Splunk as a Service”
- Splunk Enterprise preinstalled in AWS (or Azure) environments
- “Get started in as little as two days.”

Lab Exercise # 1 (30 min)

- **Lab Environment**
- **Event Generator Configuration**
- **Apps - “Destinations”**
- **Quick Guided Tour**

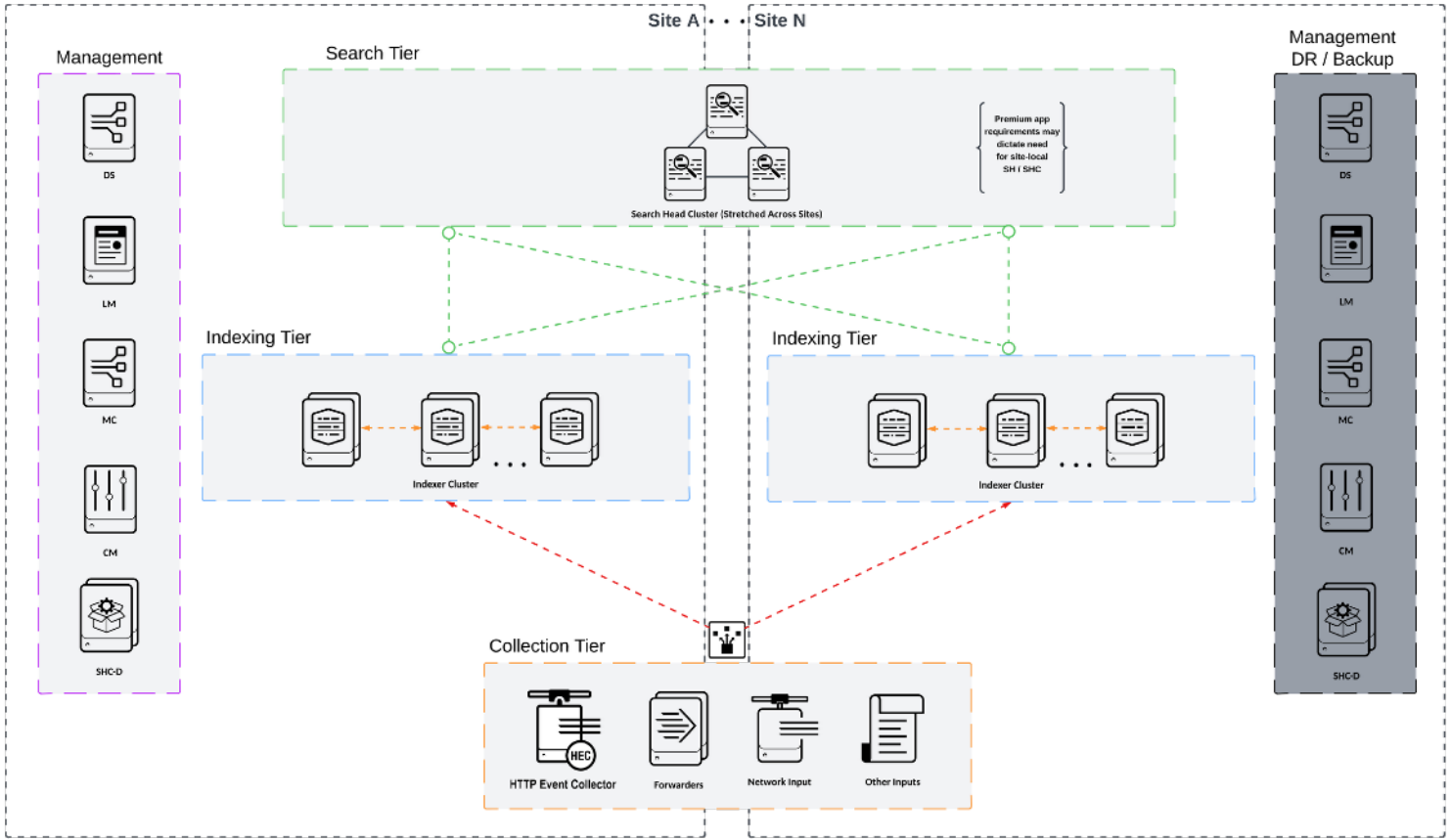
Splunk Architecture

Part #2

Components of Splunk Deployments

- **Standalone (simplest and default)**
  - **Indexing**
  - **Searching**
- **Components**
  - **Indexer = storage (replication, cluster, search peers)**
  - **Search head = clustering for availability**
  - **Forwarder = send data to indexer**
  - **Deployment server = distributing configurations, apps, add-ons**

Sample Cluster



Dimensions of a Splunk Architecture

- Amount of incoming data (process time)
- Amount of indexed data (I/O bandwidth)
- Number of concurrent users (resources)
- Number of saved searches (capacity)
- Types of search you use
- Whether or not you run Splunk Apps

Indexes

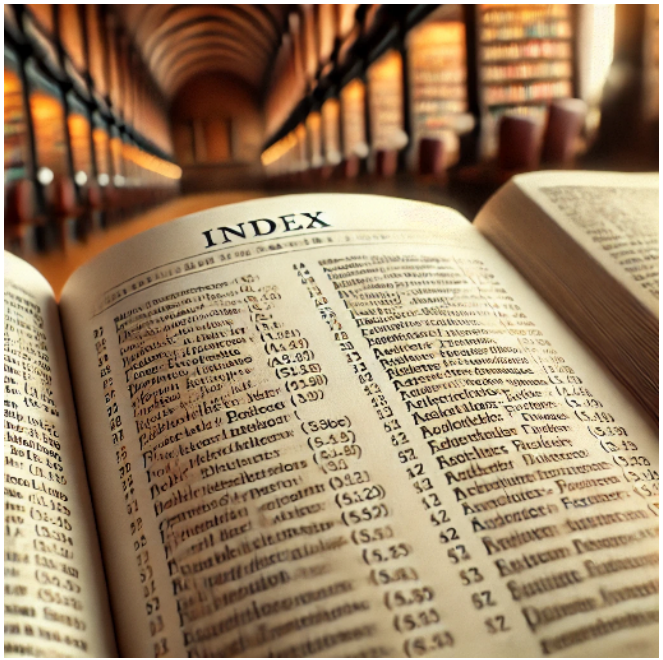
Part #3

Splunk Data

- Streaming data (real-time)
- Analysis of data (offline)
- Data Sources
  - Machine data
  - Sensor Data (IoT)
  - Weblogs
  - Social media
  - Any data - It will be timestamped

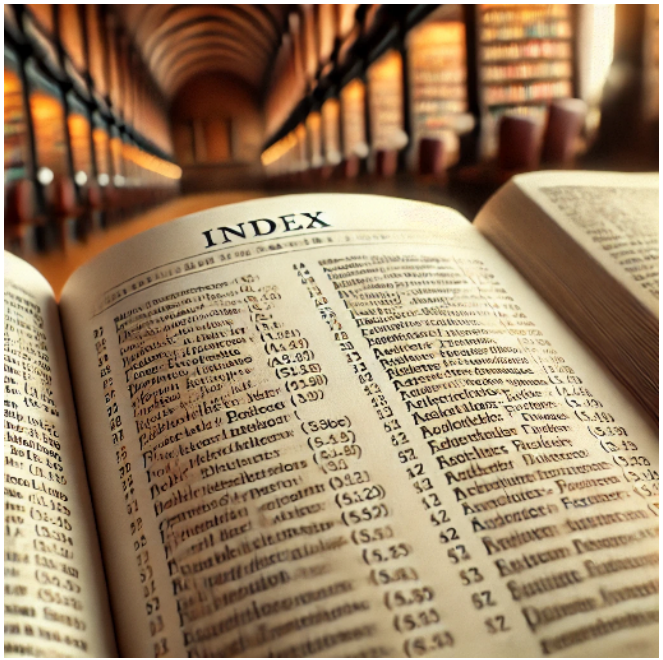
Index - Splunk Data Repositories

- **Index**
  - Data repository
  - Events are stored in indexes
- **Indexer - Instance of Splunk Storage**
  - (Potentially) Parse incoming data into events
  - Index the data
  - Participate in Search
- **Indexer Cluster**
  - Group of indexers
  - Replication of data (highly-available)
  - Clustering (parallelization for performance)



Index Content

- **Compressed raw parsed data and metadata**
- **Age-designated index directories**
- **Index directories are known as buckets**
  - Retirement and archiving policies
  - Bloom Filters
  - Time-series Indexes
- **Flat files, no third-party databases**



Bucket Aging States

Bucket State / Age	Description	Searchable?
Hot	Contains newly indexed data. Open for writing. One or more hot buckets for each index.	Yes
Warm	Data rolled from hot. There are many warm buckets. Data is not actively written to warm buckets.	Yes
Cold	Data rolled from warm. There are many cold buckets.	Yes
Frozen	Data rolled from cold. The indexer deletes frozen data by default, but you can choose to archive it instead. Archived data can later be thawed.	No
Thawed	Data restored from an archive. If you archive frozen data, you can later return it to the index by thawing it.	Yes

Exercise # 2 (15 min)

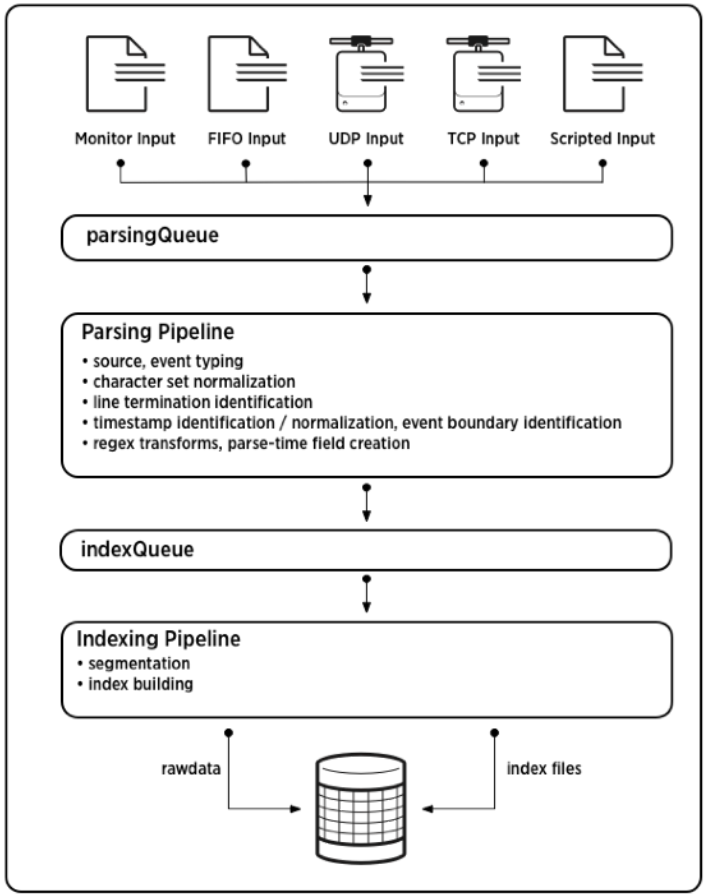
- **Create an Index**
- **Monitor log files**

Apps and Add-Ons

- Extend functionality / User Access Inflection Point
- Apps
  - Self-created
  - Paid & Free
  - Might use one or more add-ons
- Add-Ons
  - Getting data in (sourcetypes)
  - Useful Data Sets (map data, tax tables, lookups)
  - Pre-built Constructs (searches, data models, reports, dashboards, alerts)
  - Visualizations
  - New commands
  - Arbitrary code

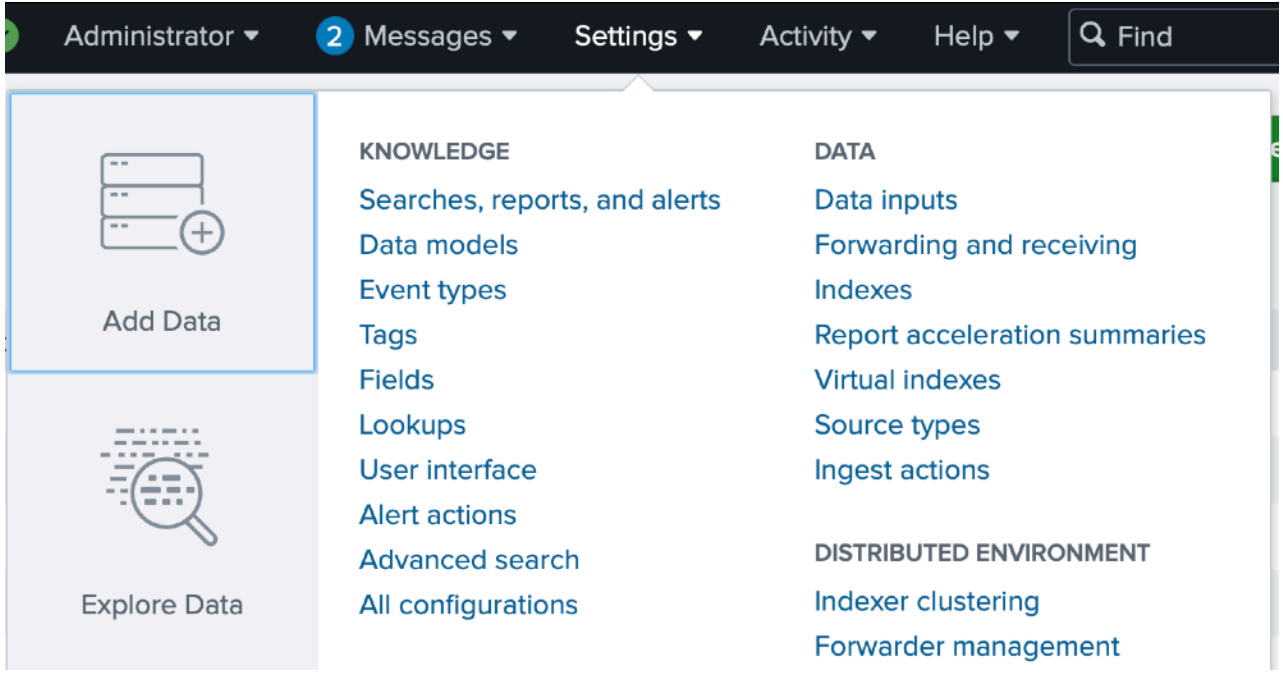
Indexing Pipeline

- Parsing
  - Extract default fields
  - Character encoding
  - Identify line breaks
  - Identify timestamps
  - Mask sensitive data
- Indexing
  - Segmentation of data
  - Build index data structures
  - Write data to disk



Lab Exercise # 3 (15 min)

- **Upload data manually**



Indexes

- Events
- Metrics
- Fields
- Fields Extraction

Events

- Set of values associated with a timestamp
- Single entry of data, one or multiple lines
- Interesting Event examples
  - Denormalized record for cross-table search
  - Text document
  - Configuration file
  - Stack trace

Events

- A web application server event:
  - 173.26.34.223 - - [01/ Mar/ 2015:12:05:27 -0700] "GET / trade/app? action=logout HTTP/1.1" 200 2953

Events Index Type

- Data is indexed as events
- Events have fields
- Default fields that all events have:
  - index - used to direct searches
  - _time - parsed timestamp
  - host - name of the device (often)
  - source - name of the file, directory, data stream
  - sourcetype - well-known types or custom
  - _raw - raw parsed event data used for extractions

Metrics Index Type

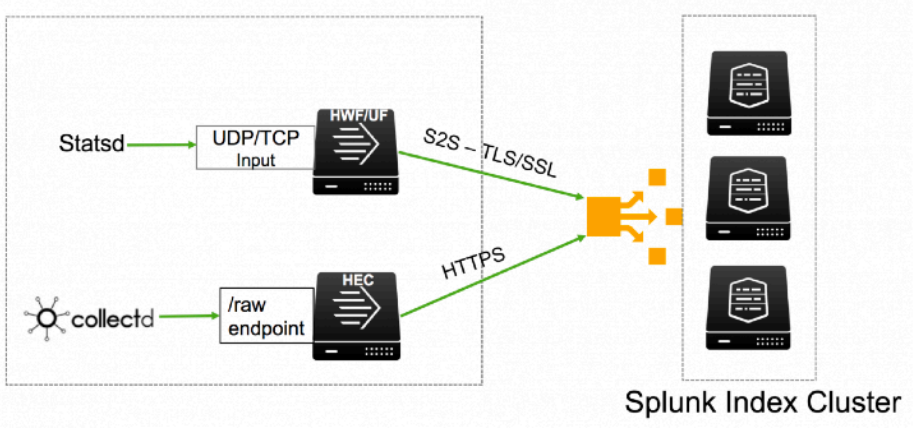
- Introduced with version 7.0 (October 2017)
- Significantly improves performance of numerical statistical analysis
  - Machine / Sensor data
  - Business metrics
- Each metric contains:
  - _time - timestamp when the measurement(s) taken
  - dimension0 - dimensionN – An arbitrary number of pieces of dimensional metadata used to differentiate metrics taken at the same timestamp (machine ID, server IP, region, department, etc.)
  - metric_name:<name>=<value> – a type of data represented, along with its value
    - metric_name:os.cpu.user=42.12345
  - There can be multiple metrics stored for any given timestamp/dimensions combination
  - Stored as a 64-bit floating point number, precision between 15 and 17 decimal digits.
  - index, source, sourcetype, host just as in Events Indexes.
- Stored in a different index type than events
  - Requires special commands for many operations (mstats, mchart, etc.)

Metric Example

_time=01/Aug/2017 12:05:27
metric_name:os.cpu.user= 42.12345
metric_name:os.cpu.idle=17.64211
dimension0:region=us-west-1
dimension1:ip=23.62.1.125

Ingesting Metrics

- statsd – plain or with dimension extensions + UDP/ TCP
- collectd + HTTP Event Collector
- metrics CSV



Event Fields

- Each individual event or metrics entry (and hence individual search results) are a collection of fields
  - Fields are searchable name/value pairs
  - Most are derived at search time:
    - Extracted from _raw field
    - Looked Up from other sources
    - Generated by commands
  - Some may be stored in the index

Field Extraction

- Some fields are automatically always indexed and available:
  - _raw – copy of parsed raw event data
  - host, source, sourcetype, linecount, timestamp, and split time values like date_year
- Sometimes fields are noticed by Splunk (Patterns Tab)
- Custom Fields at Search time
  - Based on known sourcetype or csv with headers
  - Using rex command
  - Using Field Extraction Tool
- Custom Fields at Index time
  - more work + more data = reduced index-time / search time performance
  - Some special cases:
    - search foo != bar, but foo is almost always = bar
    - search foo=bar, but most events have foo != bar yet have term bar in them elsewhere

Lab Exercise # 4 (10 min)

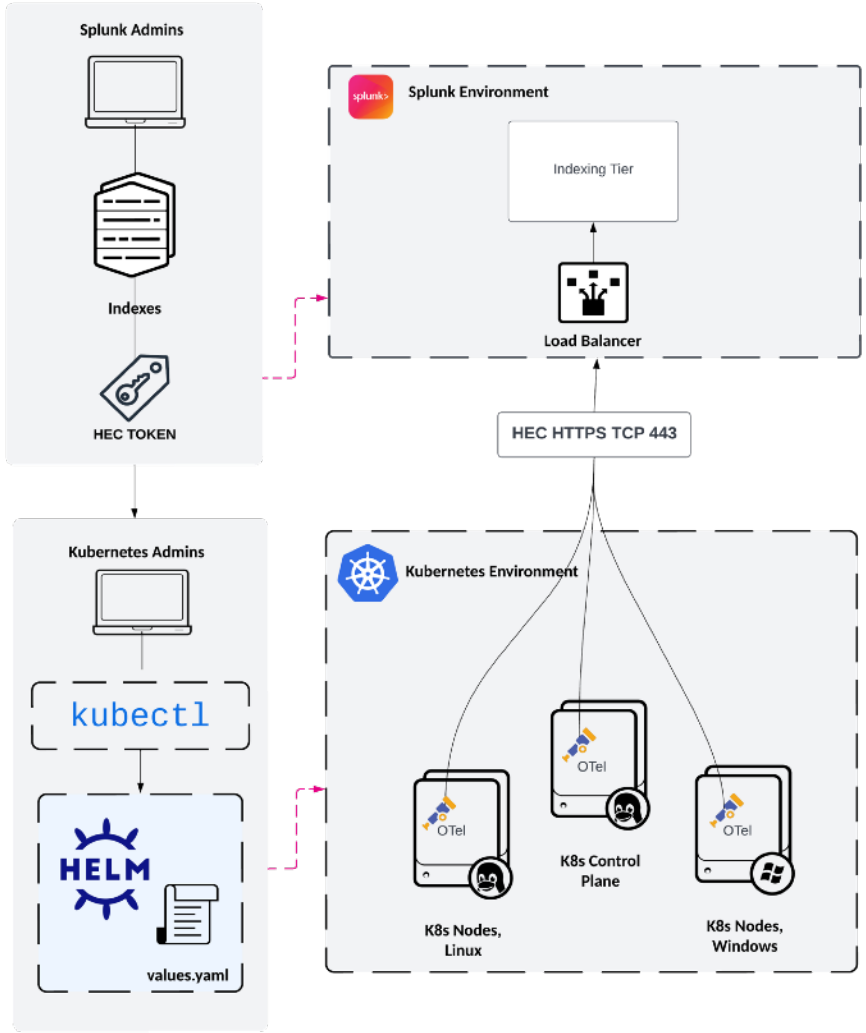
- **Field Extractor Tool**

Managing Data

- Forwarders
- HTTP Event Collector
- Removing Data

Forwarders

- **Instances that forward data to**
  - Splunk Indexers or Intermediate Forwarders
  - Third-party systems
- **Types**
  - Non-Splunk
    - collectd, fluentd
    - Otel (Open Telemetry) – For Kubernetes infrastructures
  - Splunk Heavy Forwarder
    - Can parse (props/transforms)
    - Can act as an HEC
    - Can reprocess data
    - Can act as an intermediate tier
  - Splunk Universal Forwarder
    - Cannot parse (no props/transforms)
    - Cannot use as an intermediate: can cause runaway queues, data loss



HTTP Event Collector (HEC)

- HEC accept events over HTTP or HTTPS
- HEC forward events to another Indexer
- Receive metrics from Single Page Apps
- You need to generate a token

Removing Events

Automatically using bucket aging policies

- or -

1. **Delete events from index using “delete” command – Does not reclaim disk space**
   - **source="/fflanda/incoming/cheese.log" | delete**
2. **Reclaim disk space from one or all indexes (CLI)**
   - **splunk stop**
   - **splunk clean eventdata**
   - **splunk clean eventdata -index _internal –f**

- **Note: By default, no users have the “delete_by_keyword” capability, which is offered in the can_delete role. Even admin role users do not have the capability. Best practice is to create a special user assigned to the can_delete role to perform deletions, then log in as that user when deletions are required.**

Removing Entire Indexes

- **From the Web UI**
  - Settings->Data->Indexes
  - Delete or Disable
- **From the CLI**
  - splunk remove index <index_name>
  - splunk disable index <index_name>

Search Processing Language

Part #4

Search Processing Language 1 (SPL1)

- **Search & Reporting App**
  - **Primary way to navigate data**
  - **Web-based interface**
  - **Command-line interface**
  - **API interface**
- **It's what we do with the events in indexes**

SPL - Keywords, Fields, etc.

- Keywords to be searched for in _raw
  - failed login – This searches for events with keyword “failed” AND (implied) the keyword “login”
- Quoted phrases to be searched for in _raw
  - “failed login” – This searches for events with the exact phrase “failed login”
- Fields - Name/Value Pairs
  - user=cludwig@instantbrains.com
  - Careful: Field names are case sensitive, field values are not
- Wildcards to be searched for in _raw or in field values
  - *ailed, fail*, user=* (special case to include field in results)
- Booleans - Careful: case sensitive - must be uppercase
  - NOT, OR, (AND)
- ( ) - Grouping terms – overriding precedence
- [ ] - Sub-search
- | - Pipe - pass results to next stage in pipeline

SPL - Basic Commands

- sort
  - Sorts results by specified fields
- dedup
  - Removes duplicates (De-dupes)
- rename
  - Renames a specific field - when field names are non-helpful or fields are composed/generated
- table
  - Builds a table with the specified fields
- stats
  - Provides statistics related to results
- chart / timechart
  - Returns results in tabular form for charting
- eval
  - Calculates a new expression value

SPL - Major and Minor Breakers

- splunk needs rules around how to index, and we need to be aware of those rules
- Major Breakers
  - A character that is used to divide words, phrases, or terms in event data into large tokens. Examples of major breakers are spaces, commas, semicolons, question marks, parentheses, exclamation points, and quotation marks.
- Minor breakers
  - A character that is used with major breakers to further divide large tokens of event data into smaller tokens. Examples of minor breakers are periods, forward slashes, colons, dollar signs, pound signs, underscores, and percent signs.
- We can use TERM() to isolate specific searchable terms … but be aware of these breakers to understand how it works.

Evaluation Order of Boolean Operators

1. **Parenthesis**
2. **NOT**
3. **OR**
4. **AND**

Searching Effectively

- **Search terms are words or parts of words with wildcards**
  - **Be aware of breakers:**
    - sea **won't match** search **but will match** destinations/sea/details
    - sea* **will match them both**
- **Search terms and field values are case insensitive**
- **Field names and the boolean operators (AND, OR, NOT) are case sensitive**
- **Search terms are additive (AND implied)**
- **Regex are done differently (via commands)**
- **Numbers are parsed at search time - except for Metrics indexes**

Two Main Types of Searches and Their Output

- **Lists of Events**
  - **Investigate to learn more about individual events**
  - **Raw events output, just filters down to only those that match**
- **Summarized Information about Events**
  - **Transforming searches (statistical commands)**
  - **No raw events returned. Data is summarized and displayed as tables or visualizations**

Type of Searches

- Distributable streaming
- Centralized streaming
- Dataset processing
- **Transforming**
- **Generating**
- **Orchestrating**

Transforming

- **Orders the search results into a data table**
- **Cell values into numerical values (statistics)**
- **Are no-streaming (need all data)**
- **Transform data into data structures**
  - **Visualizations:** column, bar, line, area, pie
- chart, timechart, stats, top, rare, typer

Generating

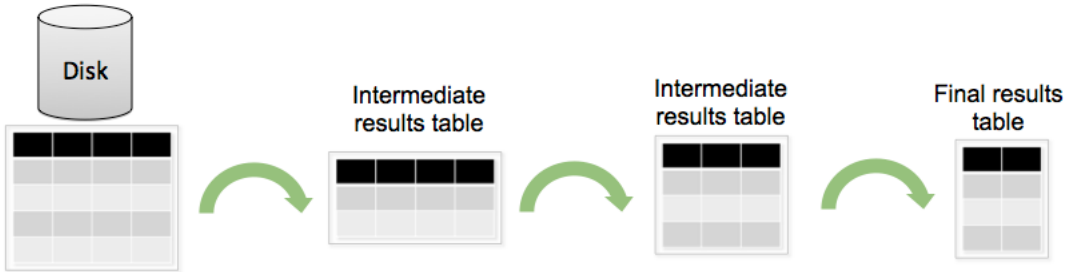
- **Fetches information, without transformation**
- **Event-generating or report-generating**
- **Results are returned in a list or table**
- **Doesn’t expect or require input**
- **Start with a leading pipe (|)**
- dbinspect, datamodel, inputcsv, metadata, pivot, search, tstats

Orchestrating

- **Control some aspect of search process**
- **It doesn’t directly affect the final result**
  - **Enable or disable a search optimization**
- redistribute, noop, localop

Anatomy of a search

- **Search = commands delimited by pipe (|)**
- sourcetype=syslog ERROR | top user | fields - percent
- **Each command refines the result data**
- **Commands are applied from left to right**



Search pipeline

- **Commands are chained together with a pipe**
- **Use output of one command (left of the pipe)**
- **Input to the next command (right of the pipe)**
- **For example**
  - Filter unwanted information
  - Extract more information
  - Evaluate new fields
  - …

Quotes and escaping

- **Quotes (") for phrases**
  - failed login
  - "failed login"
- **Quotes to ignore keywords** (error “AND”)
- **Escape special characters with backslash (\)**
  - \|, \”, \\
- **Asterisks (*) are a special character in regex/rex**
  - index=_internal | regex ".**.*"

Fields: Comparison Operators

- =
- !=
- <
- >
- <=
- >=

Wildcards (*)

- **Match unrestricted characters in a string**
  - my* => myhost1, myhost.ny.mydomain.com, myeventtype, etc.
  - *host => myhost, yourhost, etc.
  - *host* => host1, myhost3, yourhost27.yourdomain.com, etc.
- **Avoid using in the middle of a string**
- **Avoid using to match punctuation**
  - /cart* => /cart.do OR /cart/error.do
- **Performance degradation**

Time Modifiers

- Timestamp values assigned automatically
- Default time is Last 24 Hours
- You can (and should, in most cases) apply them from UI
  - A list of preset time ranges
  - Custom relative time ranges
  - Custom real-time time ranges
  - Custom date ranges
  - Custom date & time ranges
  - Advanced by using UNIX timestamps

Time Modifiers in searches

- **earliest=<time> (or starttime)**
- **latest=<time> (or endtime)**
- earliest=10/19/2017:0:0:0 latest=10/27/2017:0:0:0
- **You can specify 10m, 10s, 1h**
- **Or earliest=-2 (back two days from now)**

Further Term/Phrase Matching

- CASE() search for case-sensitive matches against terms or field values
  - CASE(“ERROR”)
  - status=CASE(“ERROR”)
- TERM() match whatever is inside parenthesis
  - **When to use? When searching for:**
    - Term may contain minor breakers
    - Is bound by major breakers (comma, space)
    - Doesn’t contain major breakers
  - **For example, TERM(127.0.0.1)**
    - **…but would not match against “ip=127.0.0.1 - user=admin ” - why?**

Regular Expressions

- **Regex are PCRE** (Perl Compatible Regular Expressions)
- **Commands: rex or regex**
  - **rex = Regular Expression eXtractions**
  - **regex = events must match pattern**
- **Pipes (|) are used to specify an OR condition in PCRE**
- **Backslash (\) to escape many interesting characters**

Lab Exercise #6 (10 min)

- **Simple Searches**
  - **Search Assistant**
  - **Basic search operators**

Searches in Splunk

- Common search commands
- CIDR Matching
- Sub-searches
-

Common Search Commands

- top
- rare
- stats
- chart
- timechart
- eval
- rex
- transaction

Command: top

- **Finds most common values**
- **Fields added to the results**
  - **count, number of events**
  - **percent, percentage of events**
- **Return a default of 50,000 results**

top [<N>] [<top-options>...] <field-list> [<by-clause>]

sourcetype=access_* | top limit=20 referer

Command: top

```
sourcetype=access_* | top limit=20 referer
```

20 Per Page ▾ Format ▾ Preview ▾

referer	count	percent
http://www.buttercupgames.com/category.screen?categoryId=STRATEGY	2284	5.777598
http://www.buttercupgames.com	1980	5.008601
http://www.google.com	1582	4.001821
http://www.buttercupgames.com/category.screen?categoryId=ARCADE	1372	3.470606
http://www.buttercupgames.com/category.screen?categoryId=NULL	1281	3.240413
http://www.buttercupgames.com/product.screen?productId=SF-BVS-G01	1210	3.060811
http://www.buttercupgames.com/category.screen?categoryId=ACCESSORIES	1082	2.737023
http://www.buttercupgames.com/category.screen?categoryId=TEE	993	2.511889
http://www.yahoo.com	766	1.937671

Command: rare

- **Displays the least common values**
- **It's like the top, but returns the opposite**

rare [<N>] [<top-options>...] <field-list> [<by-clause>]

sourcetype=access_* | rare limit=5 url

Command: stats

- **Calculates aggregate statistics over results**
  - average, count, sum, etc.
- **Statistics based on your events**

stats (stats-function(*field*) [AS *field*])... [BY *field-list*]

sourcetype=access* | stats avg(kbps) BY host

sourcetype=access*| top limit=100 referer_domain | stats sum(count) AS total

Command: stats

```
sourcetype=access_* | stats count BY status, host
```

status	host	count
200	www1	11835
200	www2	11186
200	www3	11261
400	www1	233
400	www2	257
400	www3	211
403	www2	228
404	www1	244
404	www2	209

Command: chart

- **Return results in a table format**
- **Used to display data as a chart**
- **Must specify a statistical function**

chart [<chart-options>] [agg=<stats-agg-term>]
(<stats-agg-term> | <sparkline-agg-term> | "("<eval-expression>")")...
[BY <row-split> <column-split>] | [OVER <row-split>] [BY <column-split>]]

... | chart max(delay) OVER foo BY bar
... | chart eval(avg(size)/max(delay)) AS ratio BY host user
... | chart sum(sales) BY products quarter

Command: chart

```
source="addtotalsData.csv" | chart sum(sales) BY products
quarter
```

products	QTR1	QTR2	QTR3	QTR4
ProductA	1200	1425	1300	1550
ProductB	1400	1175	1250	1700
ProductC	1650	1550	1375	1625

Command: timechart

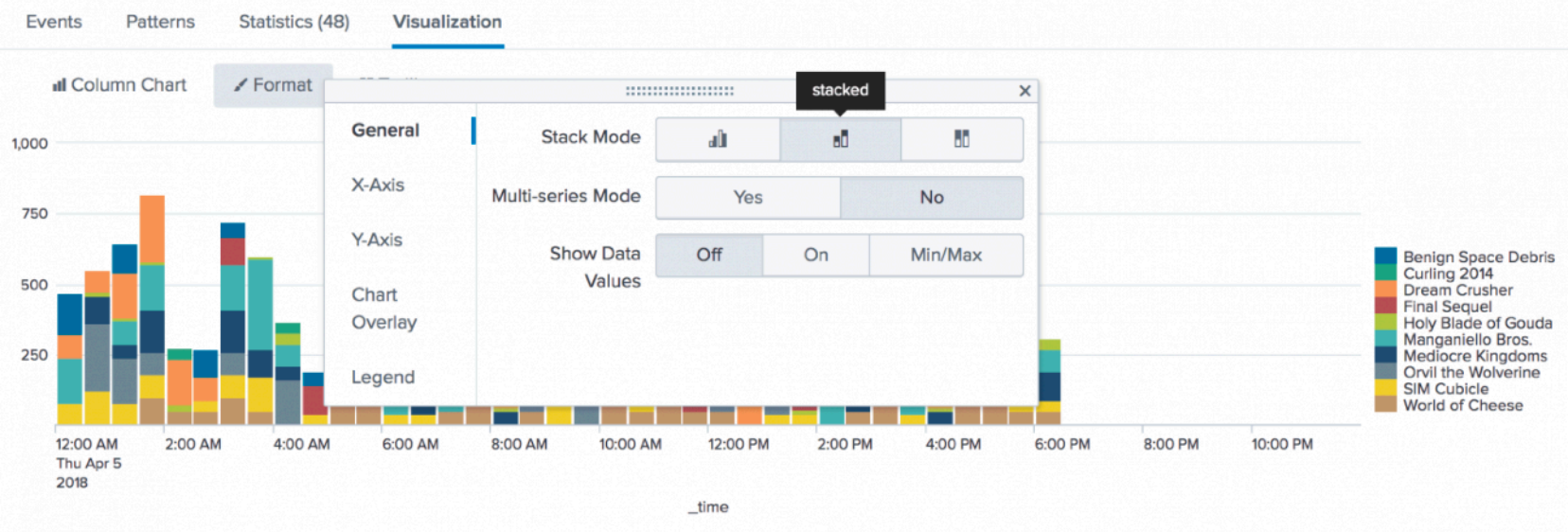
- **Create a timeseries chart**
- **Time is used as the X-axis**

timechart [sep=<string>] [format=<string>] [partial=<bool>] [cont=<bool>] [limit=<int>] [agg=<stats-agg-term>] [<bin-options>...] **(**(<single-agg> [BY <split-by-clause>]) | (<eval-expression>) BY <split-by-clause> **)**

... | timechart eval(round(avg(cpu_seconds),2)) BY processor

... | timechart span=1m avg(CPU) BY host

Command: timechart



Command: eval

- **Calculates an expression, put it in results**
  - Mathematical, string, boolean
- **Creates new fields in your events**
- **Overwrite any existing field**

eval <field>=<expression>["," <field>=<expression>]...

... | eval velocity=distance/time

... | eval error = if(status == 200, "OK", "Problem")

Command: eval + stats

Time	Event
7/18/15 6:20:56 …PM	182.236.164.11 - - [18/July/2015:18:20:56] " GET /cart.do?action=addtocart&itemId=EST-14…
7/18/15 6:21:04 …PM	182.236.164.11 - - [18/July/2015:18:21:04] " POST /oldlink?itemId=EST18&JESESSIONID=SD6SL8FF10…

Result set before stats command

… | stats count(eval(method="GET")) as GET by host

host	GET
www1	8413
www2	4654

Result set after stats command

Command: eval

```
source=all_month.csv | eval Description=case(depth<=70,
"Shallow", depth>70 AND depth<=300, "Mid", depth>300, "Deep") |
stats count min(mag) max(mag) by Description
```

Description	count	min(Mag)	max(Mag)
Deep	35	4.1	6.7
Mid	635	0.8	6.3
Shallow	6236	-0.60	7.70

Command: rex

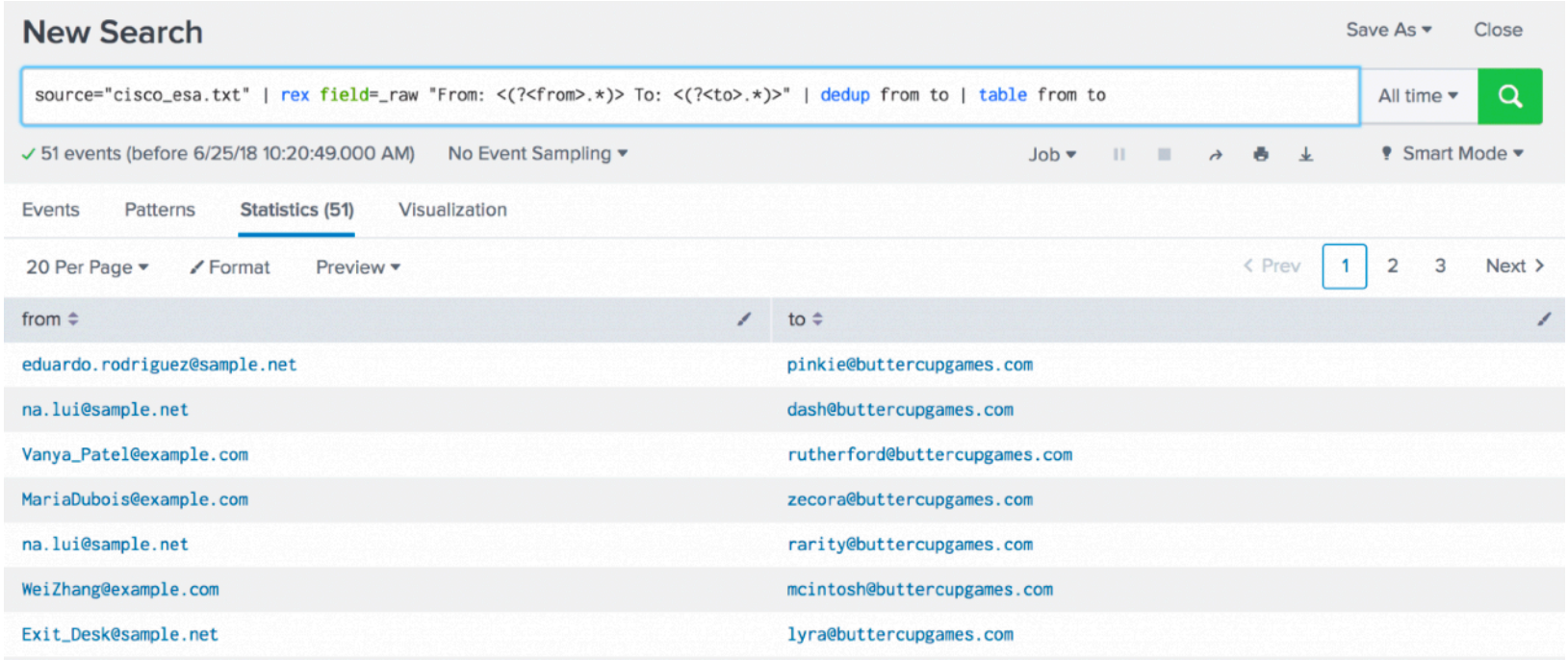
- **Extract fields using regular expressions**
- **Replace strings using the sed expression**

rex [field=<field>] (<regex-expression> [max_match=<int>]
[offset_field=<string>]) | (mode=sed <sed-expression>)

source="cisco_esa.txt" | rex field=_raw "From: <(?<from>.*)> To: <(?<to>.*)>”

... | rex field=ccnumber mode=sed "s/(\d{4}-){3}/XXXX-XXXX-XXXX-/g"

Command: rex


New Search
Save As
Close
source="cisco_esa.txt" | rex field=_raw "From: <(?<from>.*)> To: <(?<to>.*)>" | dedup from to | table from to
All time
✓ 51 events (before 6/25/18 10:20:49.000 AM)
No Event Sampling
Job
Smart Mode
Events
Patterns
Statistics (51)
Visualization
20 Per Page
Format
Preview
Prev
1
2
3
Next
from
to
eduardo.rodriguez@sample.net
pinkie@buttercupgames.com
na.lui@sample.net
dash@buttercupgames.com
Vanya_Patel@example.com
rutherford@buttercupgames.com
MariaDubois@example.com
zecora@buttercupgames.com
na.lui@sample.net
rarity@buttercupgames.com
WeiZhang@example.com
mcintosh@buttercupgames.com
Exit_Desk@sample.net
lyra@buttercupgames.com

Command: transaction

- **Find groups of related events**

transaction [<field-list>] [name=<transaction-stanza-name-from-transactiontypes.conf>] [<txn_definition-options>…]
[<memcontrol-options>…] [<rendering-options>...]

host="www.destinations.com"
| transaction client_ip endswith="confirmation"
| where duration > 0

CIDR matching

- The `search` command can perform a CIDR match on a field that contains IPv4 and IPv6 addresses.
- Suppose the `ip` field contains these values:

  10.10.10.12
  50.10.10.17
  10.10.10.23

- If you specify `ip="10.10.10.0/24"`, the search returns the events with the first and last values: 10.10.10.12 and 10.10.10.23.

Subsearches

- **A search within primary, or outer, search**
- **Subsearch is run first**
- **Enclosed in square brackets []**
- **A generating command (search, eventcount)**
- sourcetype=syslog [search sourcetype=syslog earliest=-1h | **top limit=1 host** | fields + host]

Lab Exercise #7 (30 min)

- **Common commands**
- **Complex searches**

More Search Features

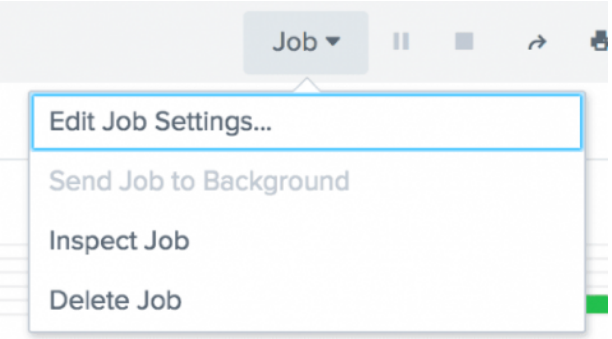
- **Interface Drilldown**
- **Manage Jobs**
- **Lookups**

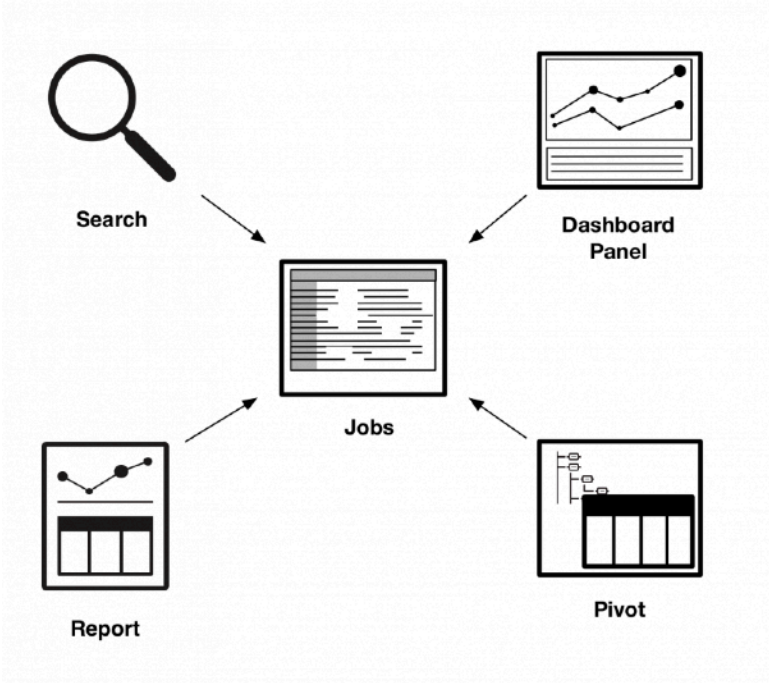
Interface Drilldown

- **Parts of the event**
  - Field values
  - Term Segments
  - Timestamps
  - Event Types
  - Tags
- **Add to search**
- **Exclude from search**
- **Remove from search**
- **New search**
- **Top values**
- **Average**
- **Timeline**

Manage Jobs

- **A Job is every time a search, pivot, report, or a dashboard panel runs**





Lookups

- Lookups enrich data
- Information for reports is elsewhere
- Add field-value combinations from lookup tables
- Type of lookups
  - CSV
  - External
  - KV Store
  - Geospatial
- Settings -> Knowledge -> Lookups

Lookups (CSV Example)

user,city,state,department

steve,Dallas,TX,HR

shelby,Dallas,TX,IT

mary,Houston,TX,HR

nanette,Houston,TX,IT

tuck,Chicago,IL,HR

Lookups

lookup [lookup definition or file name] [matching field]

sourcetype=”logs"
| lookup users.csv user
| stats count by user city state department

Defining a Lookup

- **Manually**
  - **Settings -> Lookups -> Lookup definitions**
- **Automatically (as another field)**
  - **Settings -> Lookups -> Automatic lookups**
  - sourcetype=”logs" department="HR" | top user

Lab Exercise #8 (20 min)

- Upload the lookup data
- Create a lookup definition
- Lookup automatically

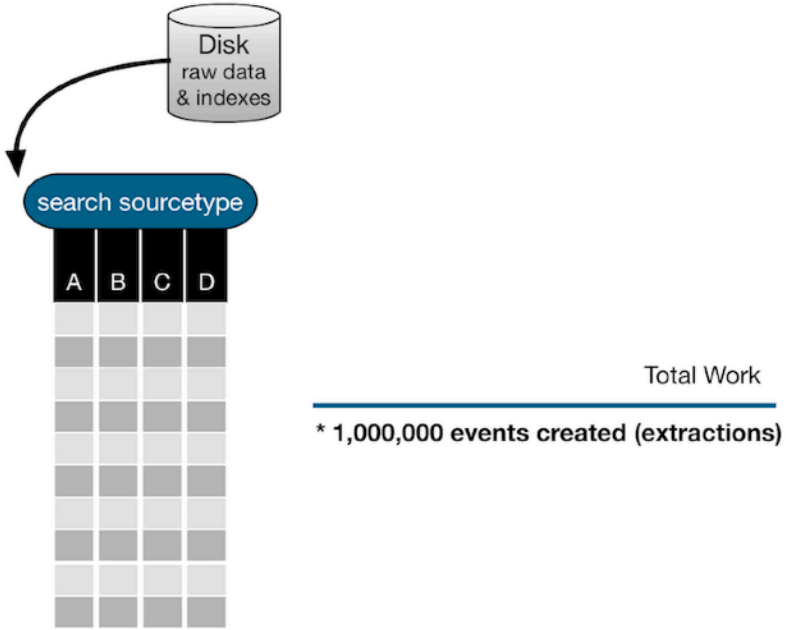
Optimizing Searches

- Prefer to use the minimum time range
- Partition data into separate indexes
- Be specific with the index name
- Be specific with the known fields
- Use as many search terms needed
- Avoid using NOT expressions
- Filter as soon as possible
- Non-streaming commands late (use all data)

We have a complex search

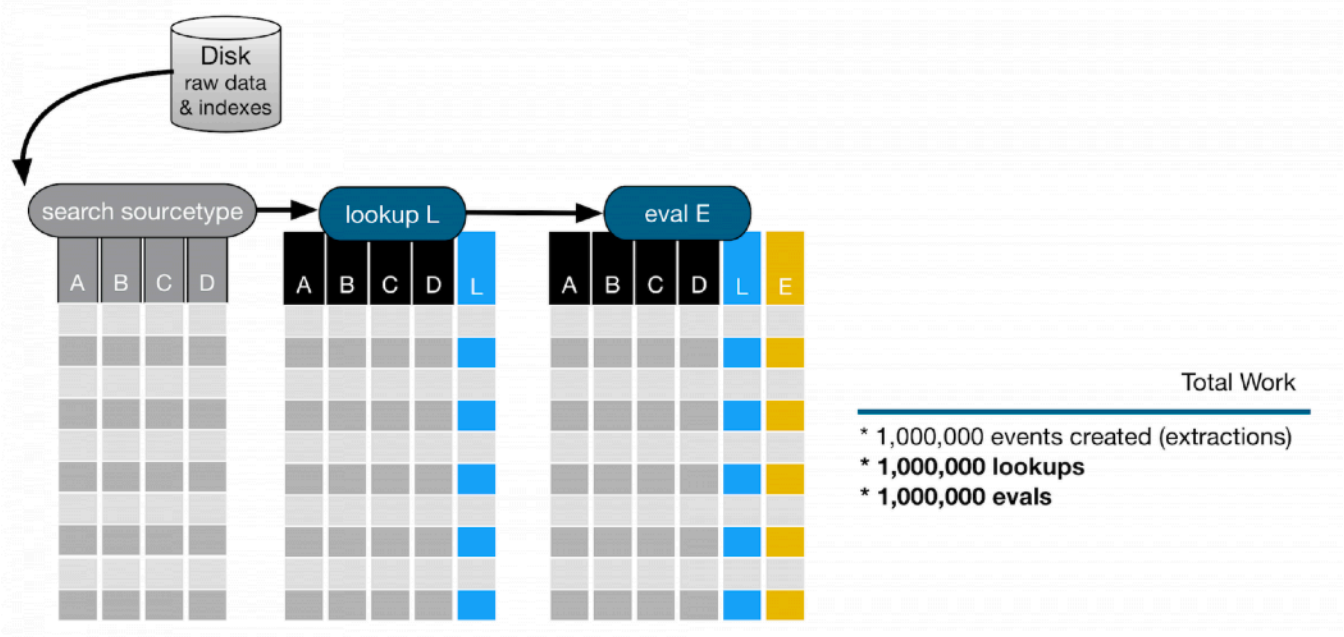
```
sourcetype=my_source
| lookup my_lookup_file D OUTPUTNEW L
| eval E=L/T
| search A=25 L>100 E>50
```

The search pipeline

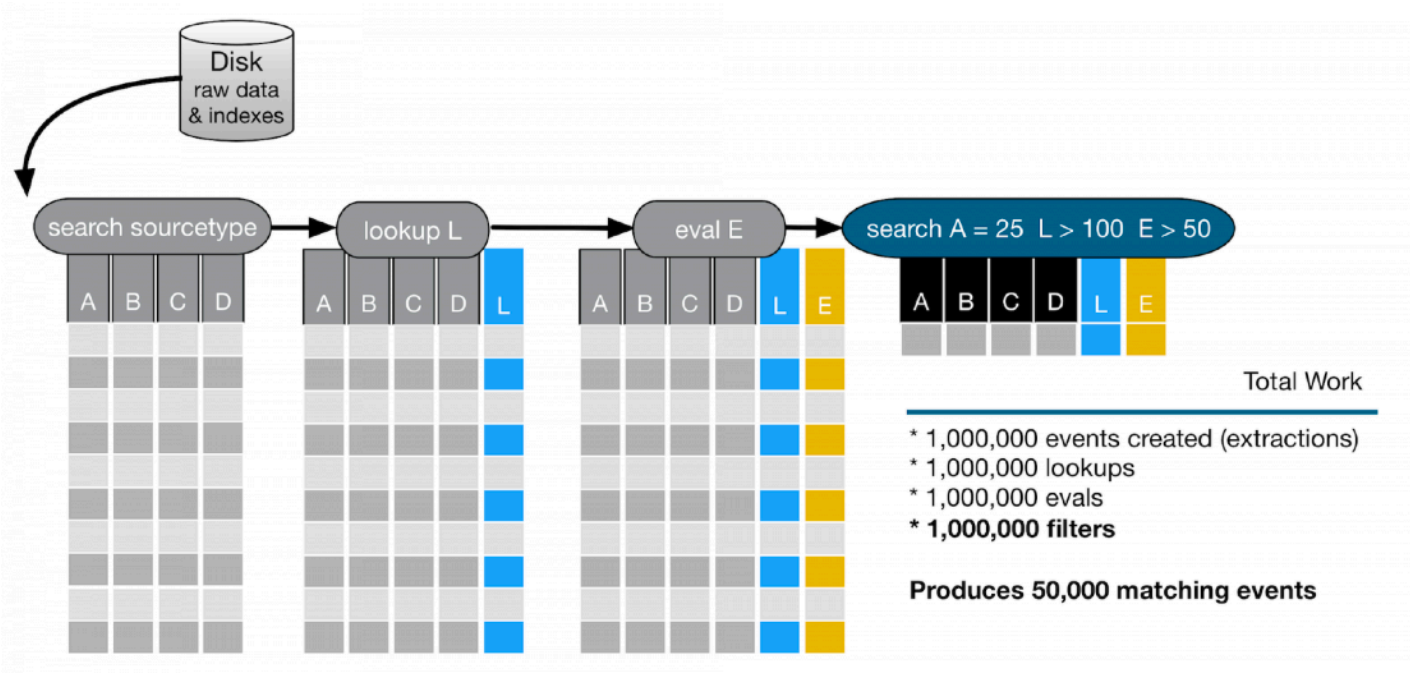


- Go to the index
- Extract 1 million events from index

1 million lookups and evals



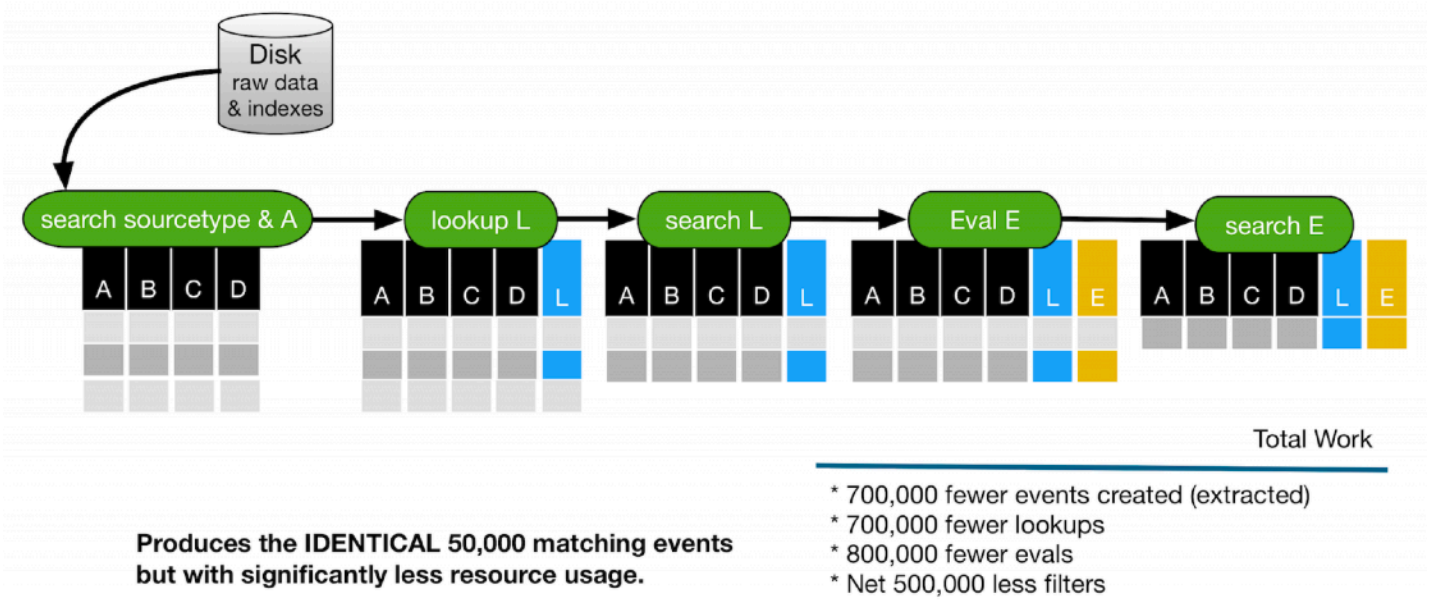
1 million filters



An optimized version

```
sourcetype=my_source A=25
| lookup my_lookup_file D OUTPUTNEW L
| search L>100
| eval E=L/T
| search E>50
```

An optimized version



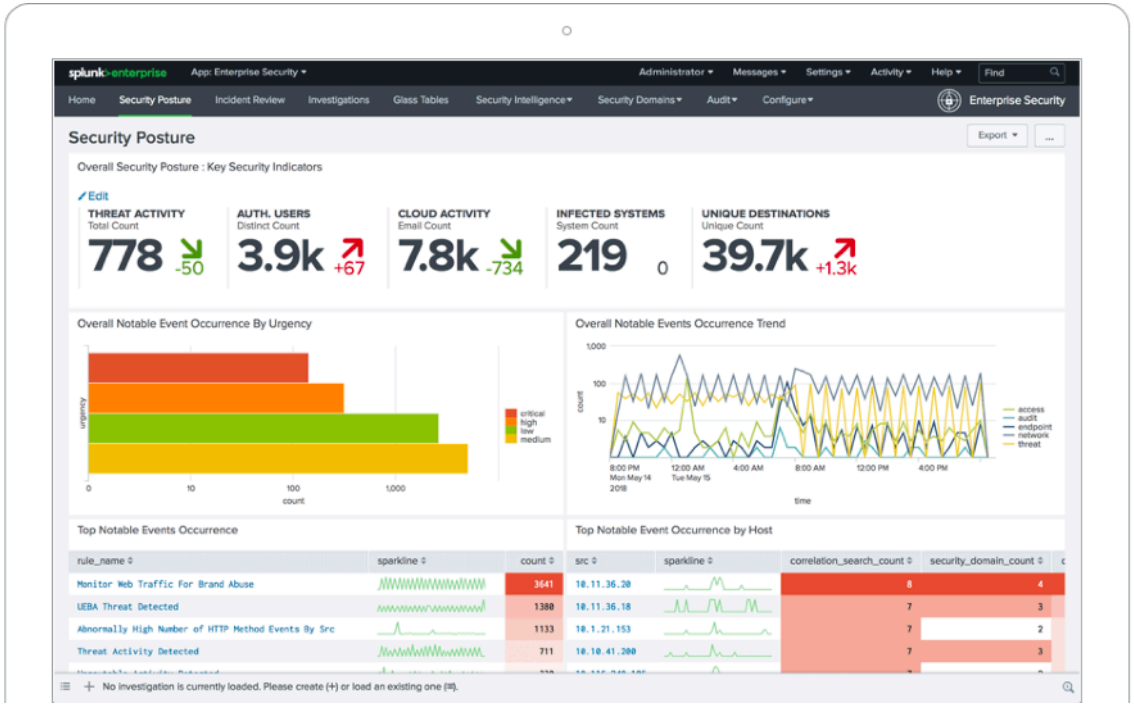
Lab Exercise #9 (30 min)

- **More advanced searches**
- **Football!**

Dashboards & Visualizations

Part #5

Dashboards in Splunk



Visualization Workflow

1. Select a visualization
2. Create a search with proper data structure
3. Configure or update the visualization
4. Create or edit a dashboard
5. Share it!

Dashboards as XML too

Visualization Types

- Events list (searches and debugging)
- Table
- Charts (pie, lines, bars, etc.)
- Single value (i.e. latency)
- Gauges (define ranges)
- Maps
- Custom

Data Structures

- **Varies by visualization type**
- **Search commands to format data**
- **Format: fields and values to present**
- **A single visualization**
  - … | stats count
- **Visualization Picker suggest searches**

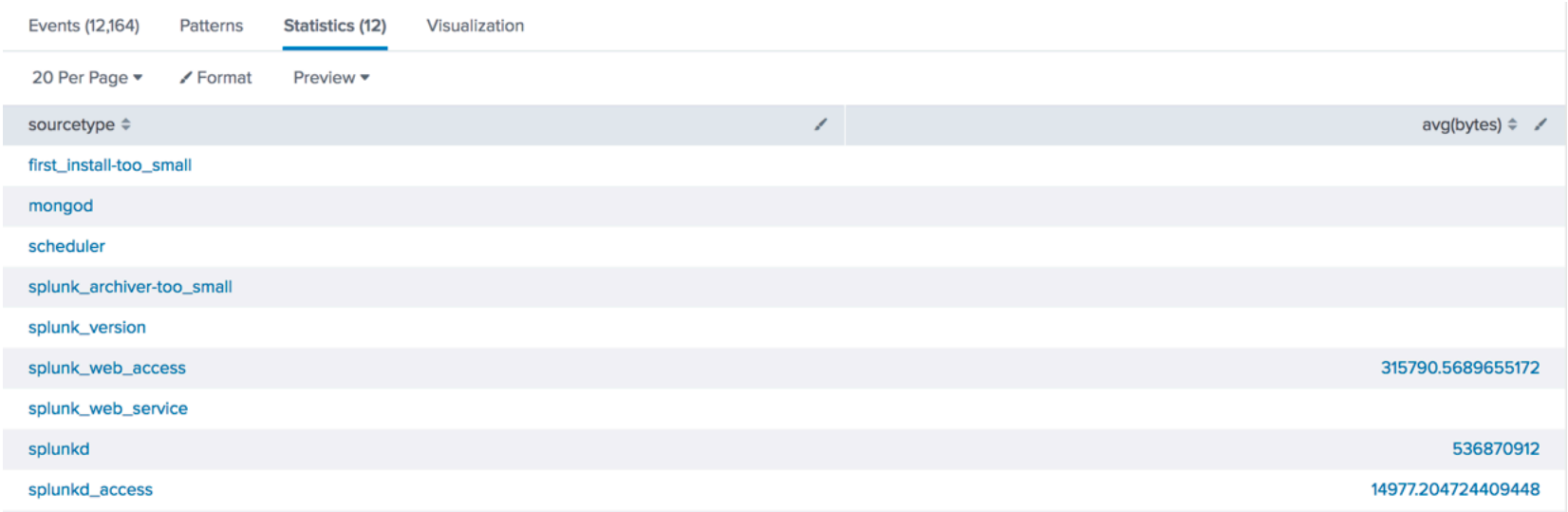
Events list

- error OR failed OR severe OR (sourcetype=access_* (404 OR 500 OR 503))

i	Time	Event
>	2/21/18 5:06:16.192 PM	02-21-2018 17:06:16.192 +0000 ERROR FromProcessor - Error in 'from' command: Invalid argument: 'error' host = so1 source = /opt/splunk/var/log/splunk/splunkd.log sourcetype = splunkd
>	2/21/18 5:06:16.192 PM	02-21-2018 17:06:16.192 +0000 ERROR FromProcessor - Error in 'from' command: Invalid argument: 'error' host = so1 source = /opt/splunk/var/log/splunk/splunkd.log sourcetype = splunkd
>	2/21/18 4:51:38.331 PM	02-21-2018 16:51:38.331 +0000 WARN HttpListener - Socket error from 10.32.31.191 while accessing /en-US/static/@B73853057830FAE7E572FD0CF4B6938B3554E8A8B0F662E419C6A5159F9D0BDC/js/common.min.js: Broken pipe host = so1 source = /opt/splunk/var/log/splunk/splunkd.log sourcetype = splunkd
>	2/21/18 4:44:00.686 PM	02-21-2018 16:44:00.686 +0000 ERROR IntrospectionGenerator:resource_usage - RU - Mount '/' () is not interesting, iostats will not be collected. host = so1 source = /opt/splunk/var/log/splunk/splunkd.log sourcetype = splunkd
>	2/21/18 4:43:58.383 PM	02-21-2018 16:43:58.383 +0000 INFO WatchedFile - File too small to check seekcrc, probably truncated. Will re-read entire file='/opt/splunk/var/log/splunk/django_error.log'. host = so1 source = /opt/splunk/var/log/splunk/splunkd.log sourcetype = splunkd

Tables

- index = _internal | chart avg(bytes) over sourcetype



Tables

- You can also apply formats
- ... | chart count(itemId) over categoryId by action

Events (106,438) Patterns **Statistics (6)** Visualization

20 Per Page ▾ Format Preview ▾

categoryId	addtocart	changequantity	purchase	remove	view
ACCESSORIES	93	43	387	43	135
ARCADE	104	51	537	57	220
SIMULATION	54	24	273	33	110
SPORTS	24	27	148	12	65
STRATEGY	167	87	885	92	344
TEE	86	25	404	39	151
	528	257	2634	276	1025

Tables with XML

```
<table>
 <search>
  <query>index=_internal | head 10000 | stats count by sourcetype</query>
 </search>
 <format type="number" field="count">
  <option name="precision">3</option>
  <option name="useThousandSeparators">false</option>
  <option name="unit">MB</option>
  <option name="unitPosition">before</option>
 </format>
</table>
```

Charts

1. **After a transformation, use the tab Statistics**
2. **Search results should have two columns**
3. **Click on then Visualization tab**

- ... | stats count by Code

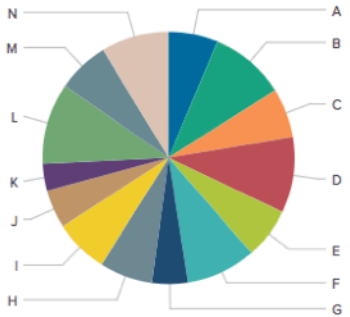


Chart: Bar

- index=_internal "group=pipeline" | stats sum(cpu_seconds) as totalCPUSeconds by processor | sort 10 totalCPUSeconds desc

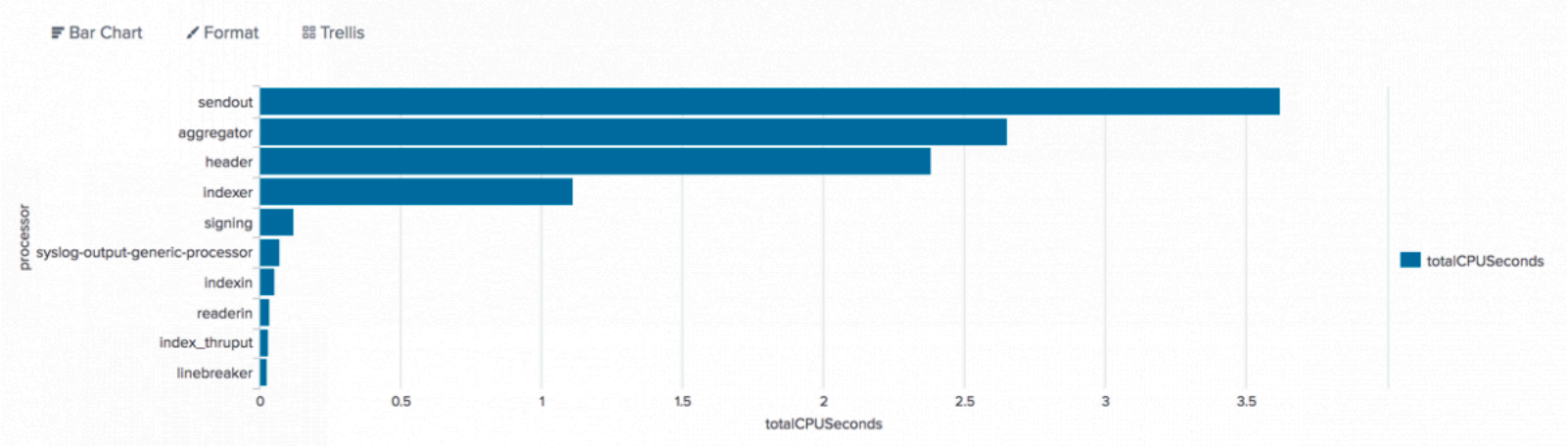
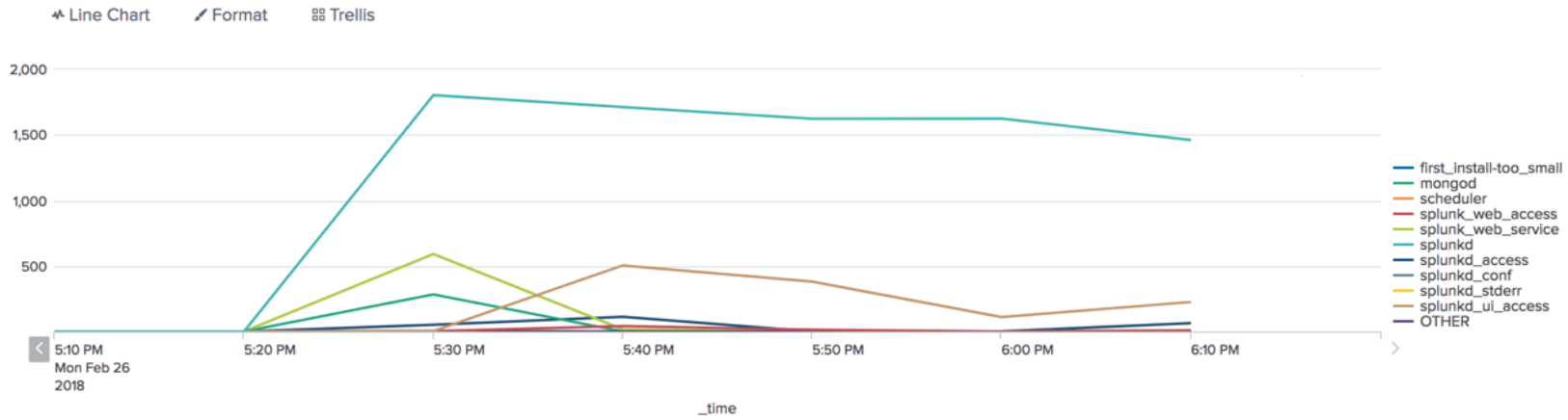


Chart: Line

- index=_internal | timechart count by sourcetype



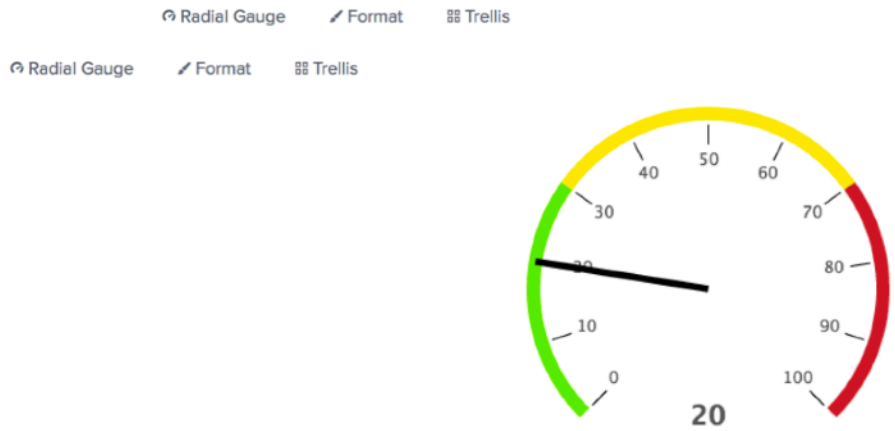
Single Value

- index=_internal source="*splunkd.log" log_level="error" | timechart count

- index = _internal source = "*splunkd.log" log_level = "error" | stats count

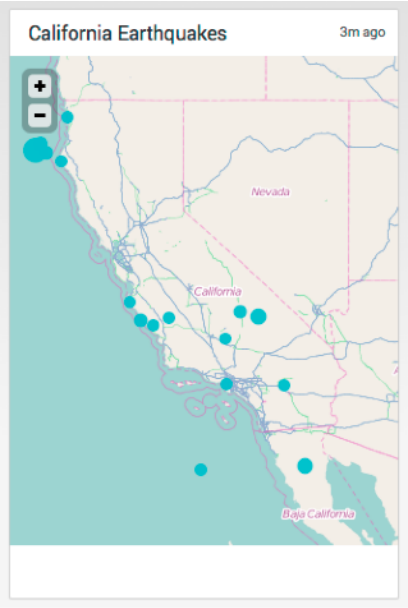
Gauges

- index=_internal source="*splunkd.log" log_level="error" | stats count as errors



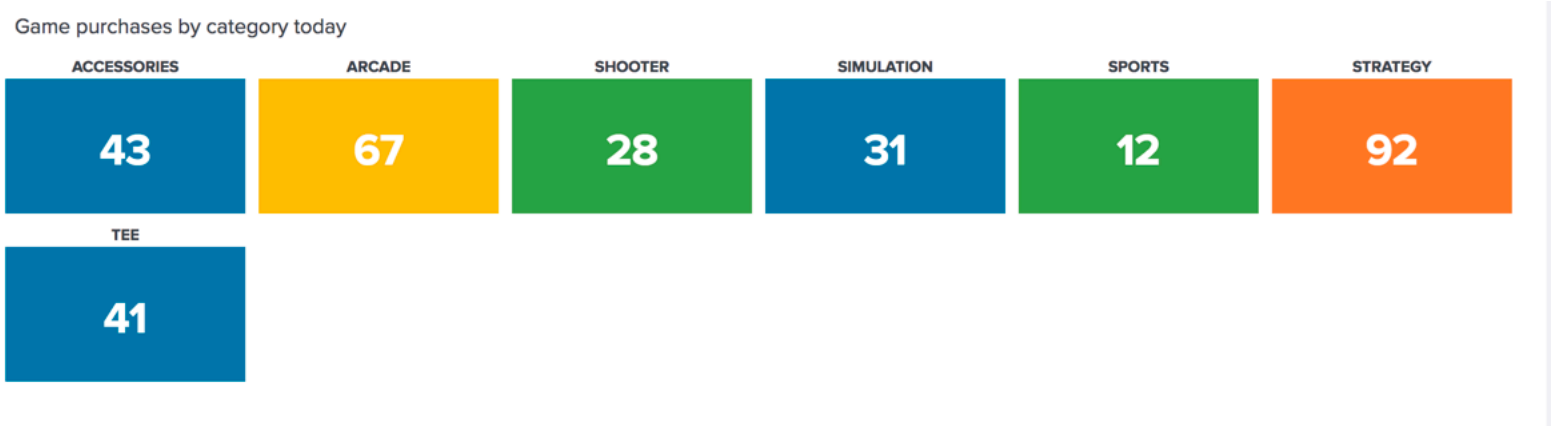
Maps

- index=main mag>3 | geostats latfield=latitude longfield=longitude count



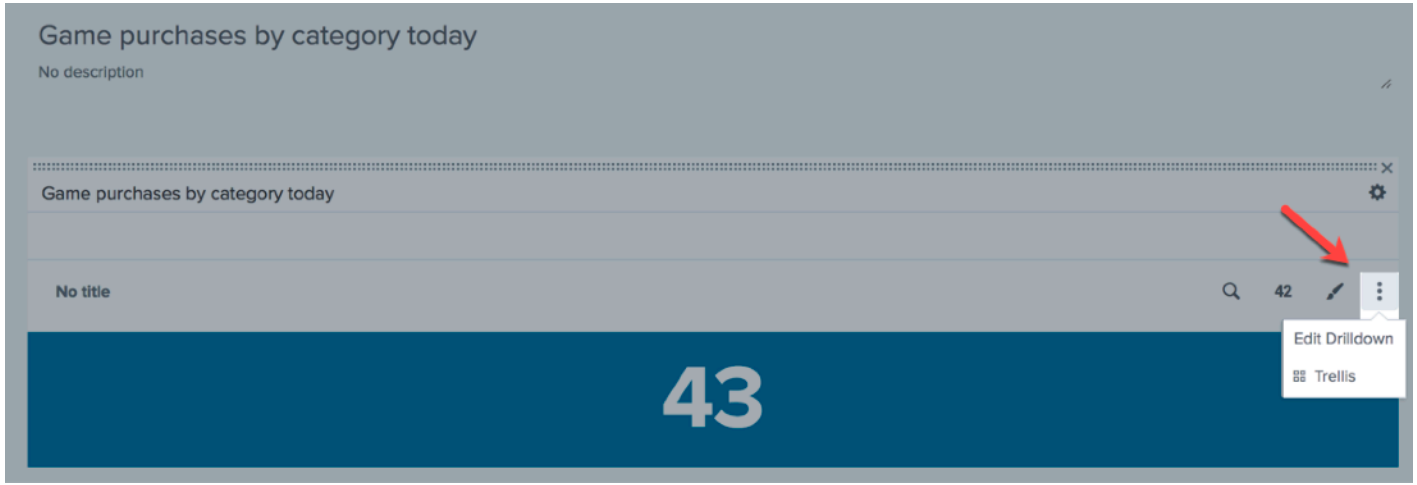
Trellis

- **Split search results**



Trellis

- Edit a dashboard
- Choose the panel
- Click “More actions” icon



Search as Report

- Save searches as visualization or report
- Save visualizations as reports or panels on dashboards
- Schedule reports (daily summary)

Dashboards

- A group of visualizations
- Represent information in a visual format
- Use layouts to structure the content
- Can be interactive
- XML is supported

Dashboards Workflow

- Create a new dashboard
- Add new visualizations
- Convert dashboards as form
- Create interactive dashboards (drilldown)
- Edit dashboards using XML (as desired)

Lab Exercise #10 (20 min)

- First Dashboard
- Saving Searches to Dashboard Panels
- Using Time Pickers on Dashboards

Interactive Dashboards

- **Inputs**
- **Drilldown**

Forms

- Similar to time series input
- Fieldsets to organize inputs
- XML
  - Root element in XML is <form>
  - Inputs in XML with <fieldset>

Forms Workflow

- Create a dashboard and add inputs
- Configure the inputs (options, behavior ...)
- Use tokens to capture selected values
- Adjust input and panel layout

Form Tokens

- **Programming variables**
- **Respond to user selections dynamically**
- **Initial search**
  - index=_internal | timechart count by sourcetype
- **Search using tokens**
  - index=_internal sourcetype=$sourcetype_token$ | timechart count

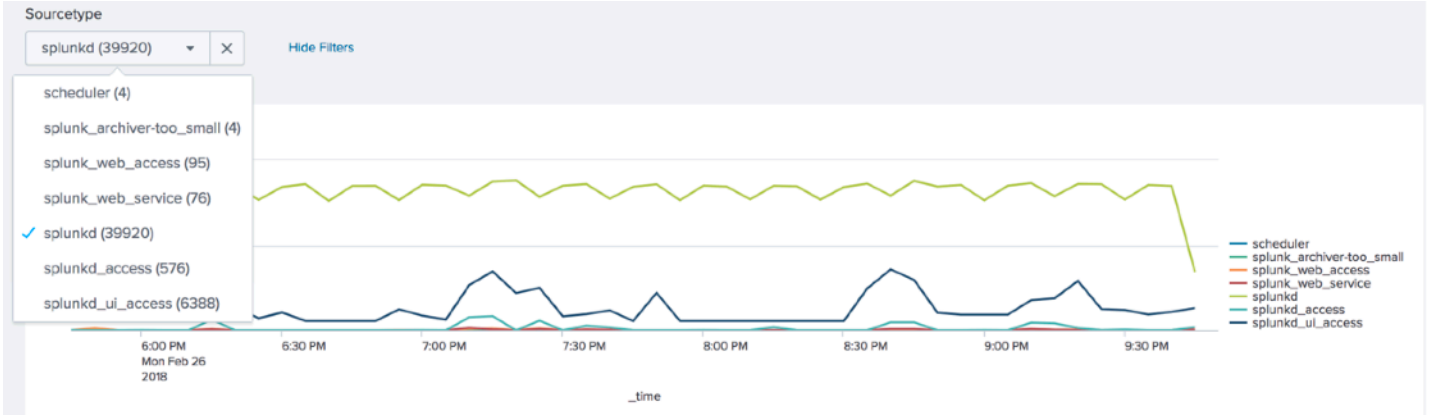
Predefined tokens for drilldown				
Predefined token	**Table**	**Chart**	**Single value**	**Map**
`$click.name$`	Leftmost field (column) name in the table.	X-axis field or category name for the clicked location	Name of field that single value represents	Field name for the clicked location
`$click.value$`	Leftmost field (column) value in the clicked table row.	X-axis field or category value for the clicked location	Field value that the single value represents	Field value for the clicked location
`$click.name2$`	Clicked table cell field name.	Y-axis field or category value for the clicked location	Same as `$click.name$`	Same as `$click.name$`
`$click.value2$`	Clicked table cell value.	Y-axis field or category value for the clicked location	Same as `$click.value$`	Same as `$click.value$`
`$row.<fieldname>$`	Access any field (column) value from the clicked table row. For example, to get the sourcetype field value in the clicked row, use `$row.sourcetype$`	Access any y-axis field value corresponding to the clicked location x-axis. Not available if the user clicks the chart legend.	Access any field value from the Statistics table row for the single value.	Access field values related to the clicked location. Check the Statistics tab for available fields.

Adding inputs

- Open the dashboard
- Click “Edit” to open the editor
- Select one or more inputs from the ”Add input” list
- Drag and drop inputs to rearrange them
- Drag an input into a specific panel (use tokens)
- Save

Lab Exercise #11 (20 min)

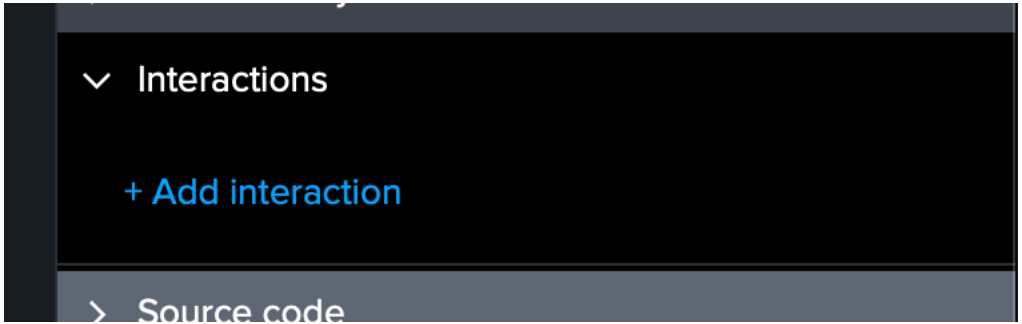
- **Add inputs**
- **Specify dynamic options**



Dashboard Interactions

- Create more interactive dashboards
- Share additional insights with users
- Trigger different interactive responses
  - Open a secondary search
  - Open another dashboard / tab
  - Open an external URL

Interactions Editor



Drilldown Tokens

- Form input change events
- Search events
- Tokens set on page load
- Chart navigation and selection events
- Predefined click event tokens
- Custom tokens

Lab Exercise #12 (15 min)

- Link to a search
- Link to a dashboard

Alerts

Part #6

Alerts

- Monitor for and respond to specific events
- Use a saved search to look for events
  - Real-time
  - Schedule
- Alerts trigger when conditions met

Alerts Workflow

- Create and save a search
- Configure how often the search runs
- Set trigger conditions
- Alert actions, i.e. notify through email

Alert Types

- **Scheduled (cron expressions)**
- **Real-time (per-result)**
  - Triggers every time there's a result
- **Real-time (rolling time window)**
  - Trigger based on result or result field counts
  - Ten results in a five-minute window

Scheduled

- A daily goal of 500 sales
- Schedule alert to search sales event at 11pm
- Configure the trigger if # results < 500

Real-time Per-result

- To know whenever login failure occur
- Search for failed login attempts
- Choose a per-result to track failed logins

Real-time Rolling Time Window

- Three failed logins in three minutes
- Search for failed logins
- Configure a rolling ten minute time window
- Throttle alert so it only triggers once per hour

Create a scheduled alert

1. Create a search
2. Save As > Alert
3. Enter title and description
4. Specify permissions
5. Configure alert scheduling
6. Configure trigger conditions
7. Configure throttling period
8. Select actions
9. Save

Recommendations

- **Search time range shouldn’t be lower than scheduled alert (miss events)**
- **Schedule alerts with one minute delay**
- **For Example**
  - Earliest: -90m
  - Latest: -30m
  - Cron Expression: 30 * * * * (every hour at min 30)
    - Runs at 4:30, events from 3 to 4
    - Runs at 5:30, events from 4 to 5

Throttle alerts

- **Suppress alert trigger for a period of time**
- **Configurations**
  - **Schedule**
    - Once (greater than zero)
    - Per-result (events with same value for fields)
  - **Real-time**
    - Rolling time window (greater than zero)
    - Per-result (events with same value for fields)

Alert actions

- **Respond to triggered alerts**
- **One or more actions**
- **Types**
  - Email
  - Webhook
  - CSV Lookup
  - Log events to another Splunk instance
  - Monitor triggered alerts
  - Custom

Email Notifications

- **Configure email notification**
  - Server > Server settings > Email settings
- **list_settings permission**
- **Tokens (placeholders) can be used**

Webhook Alert Action

- Callbacks to web resource
- HTTP POST + JSON
- Payload includes
  - Search Saved ID
  - Link to search results
  - Search owner and app
  - First result row

CSV Lookup

- **Write results to a CSV lookup**
- **outputlookup command**
- **Use case**
  - Save events with error for delayed processing
  - Events that might need a format change
  - Hosts that are having problems

Lab Exercise #13 (5 min)

- Create a scheduled alert
- Create a scheduled alert using cron
- Create a real-time per-result alert
- Create a real-time rolling window
  - (Don’t choose “per-result”)

Scheduled Reports

Part #7

Schedule Reports in Splunk

- A report that runs on a scheduled interval
- Can trigger an action each time it runs
  - Send a report by email
  - Write report to a CSV file
  - Set up a Webhook
  - Log and Index searchable results
- It’s for saved searches and dashboards
- You can embed reports to be used externally
  - Displays the results from last run
- You can set priorities for concurrent reports

Creating a Schedule Report

1. Save a search as a report
2. On the Schedule line, click Edit
3. Select the schedule for the report
4. Select the time range
5. Add Actions
6. Save

Add new

Searches, reports, and alerts » Add new

Destination app *

Search & Reporting (search)

Search name *

Purchased products, last 24 hours

Search *

```
sourcetype="access*" action="purchase" | stats
count by product name
```

Description

Run as

(•) Owner ( ) User

Learn more

Time range

Start time

-24h

Finish time

Time specifiers: y, mon, d, h, m, s

Learn more

Acceleration

[x] Accelerate this search

Summary range

1 Month

Schedule and alert

[ ] Schedule this search

Cancel

Save

Lab Exercise #14 (5 min)

- **Schedule report from a search**
- **Schedule report from a dashboard**

Questions

info@agilebrainsconsulting.com

About Agile Brains Consulting:

ASSESSMENT

Strategic analysis of the current state of IT, Business & Operations teams to provide a 360 comparison against industry benchmark and make SMART recommendations

TRAINING

Upskilling of individuals and teams with certification and non-certification, customizable workshops in professional development and modern technical tools

PROCESS TRANSFORMATION

Developing self sustaining Lean Agile & DevOps capabilities to drive business and operations in a cost-effective, timely manner

COACHING

Empowerment of people by mentorship, teaching, guidance to enable and extract their best abilities and improve performance

TECHNICAL EXECUTION

Product / project management coupled with technical solution implementation to deliver cutting edge business

DIGITAL TRANSFORMATION

Human centered, design thinking approach to digitizing and automating current processes and tools to gain business intelligence

Engagement Lead – Abrar Hashmi (reach out to ahashmi@agilebrainsconsulting.com)

THANK YOU !!