splunk> .conf2017

A Trip Through The Splunk Data Ingestion And Retrieval Pipeline

Harold Murn | Senior Systems Engineer

2017-09-27 | Washington, DC

Forward-Looking Statements

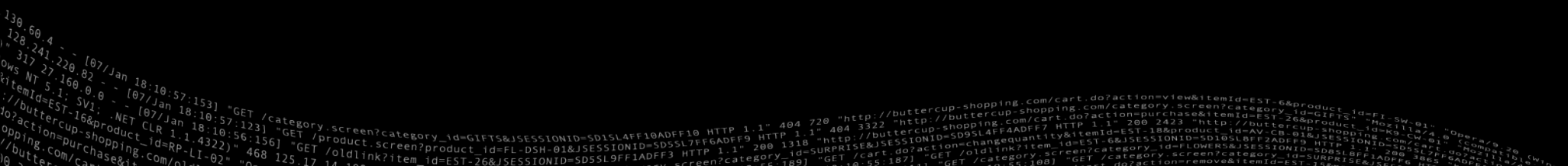
During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

splunk> .conf2017

Agenda

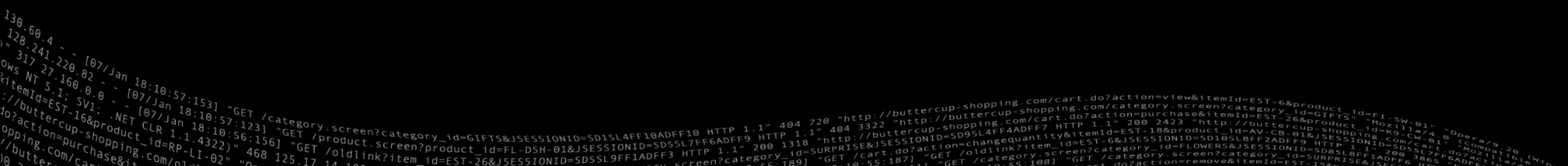
- Disclaimer
- The Search Funnel
- A Brief Overview Of Splunk Indexes
- The Easy Stuff
- What's In A Bucket?
- Bloom Filters
- Segmenting
- Time-series Indexes
- Takeaways



splunk> .conf2017

Disclaimers

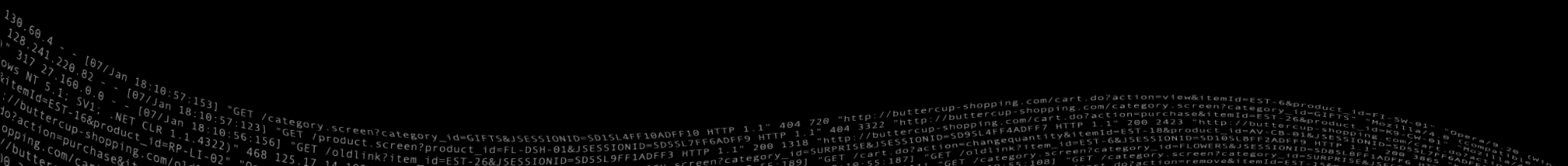
- I am not here on behalf of my employer
- The opinions and work are my own
- This largely applies only up to the first |
- I am not good with PowerPoint



Disclaimers

- I am not here on behalf of my employer
- The opinions and work are my own
- This largely applies only up to the first |
- I am not good with PowerPoint

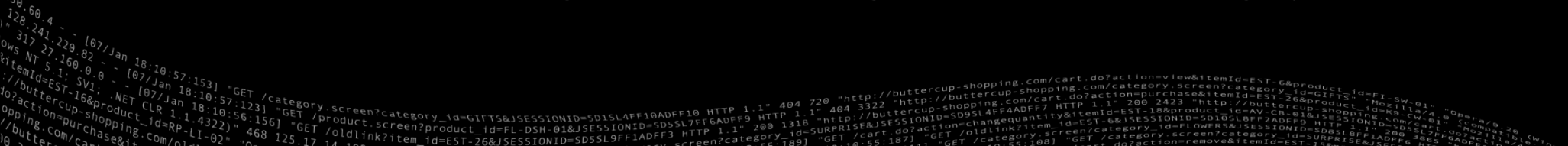
- The code will be indicative / demonstrative
- The code will be slow
- The code will be bad, but readable
- There will be no math



Disclaimers

- I am not here on behalf of my employer
- The opinions and work are my own
- This largely applies only up to the first |
- I am not good with PowerPoint

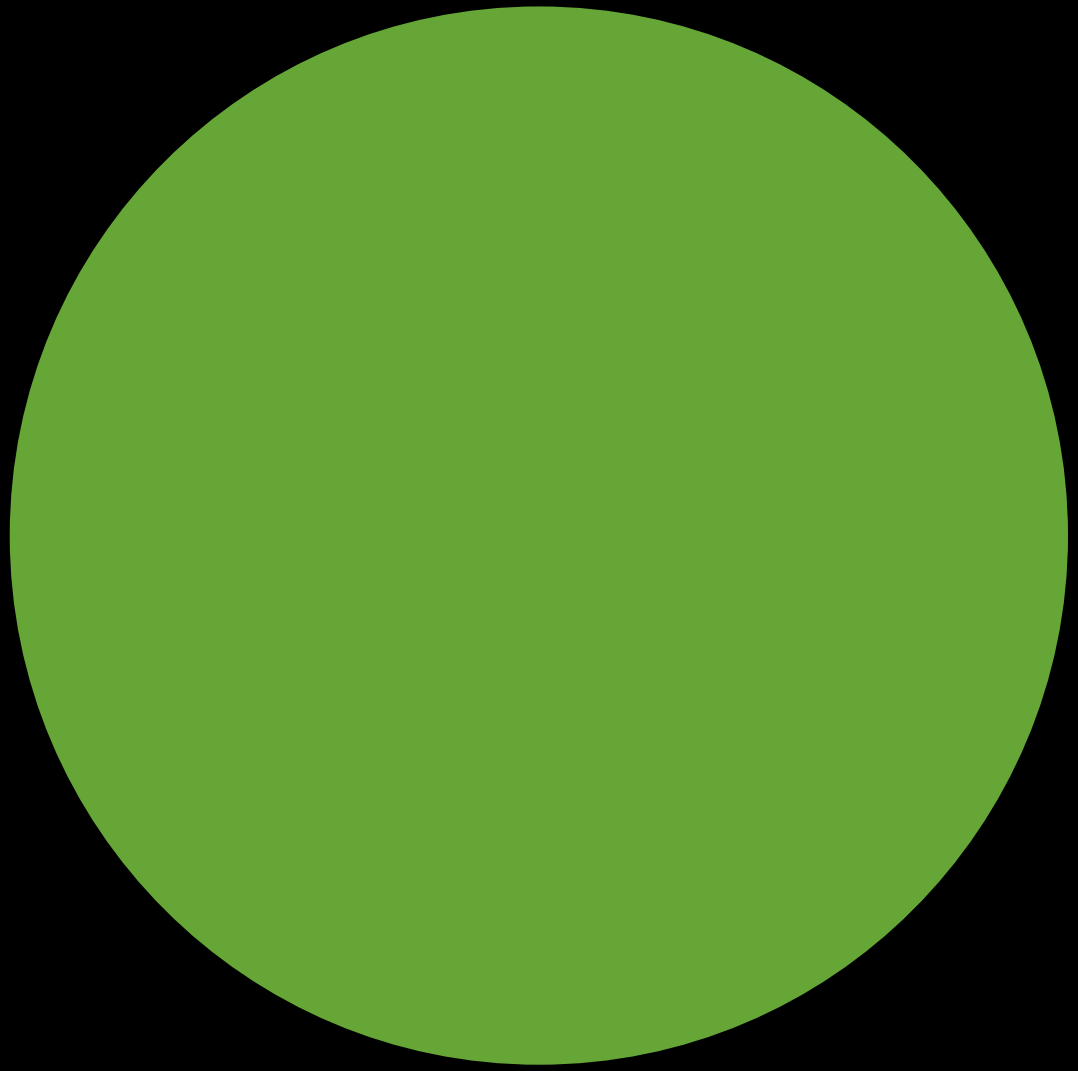
- The code will be indicative / demonstrative
- The code will be slow
- The code will be bad, but readable
- There will be no math
- The code will be on github! https://github.com/tiedotguy/conf2017



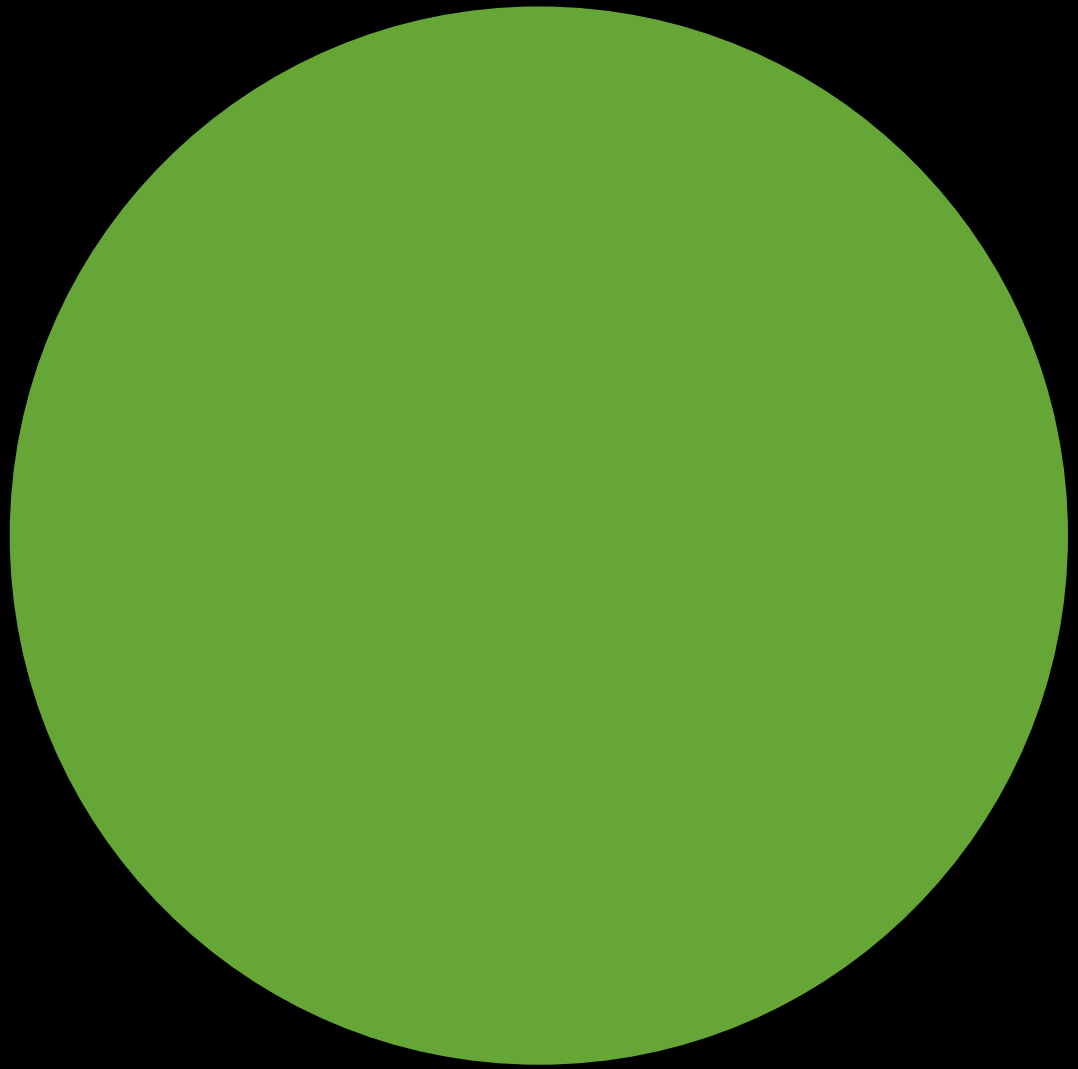
The Search Funnel

splunk> .conf2017

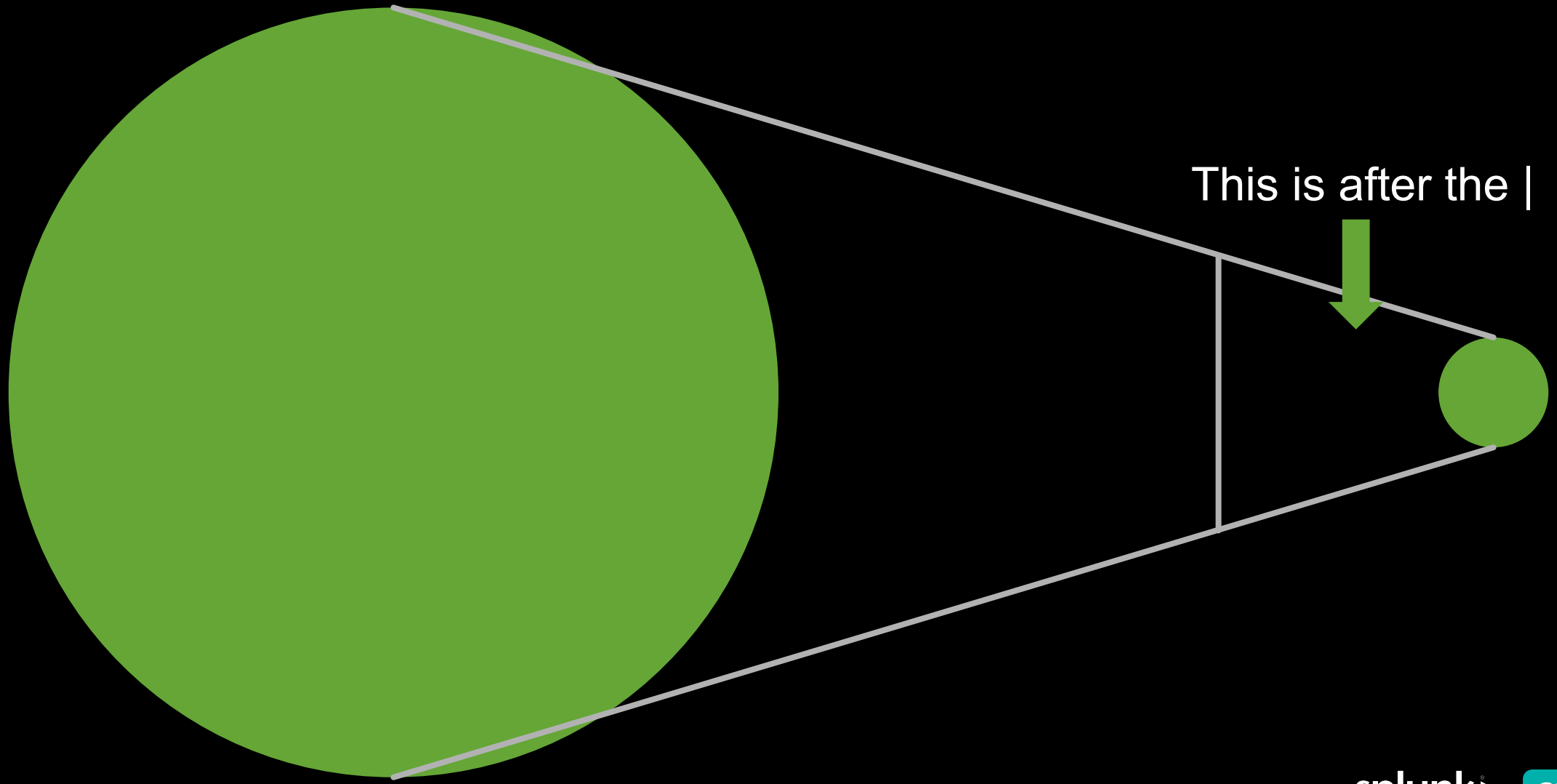
The Search Funnel



The Search Funnel

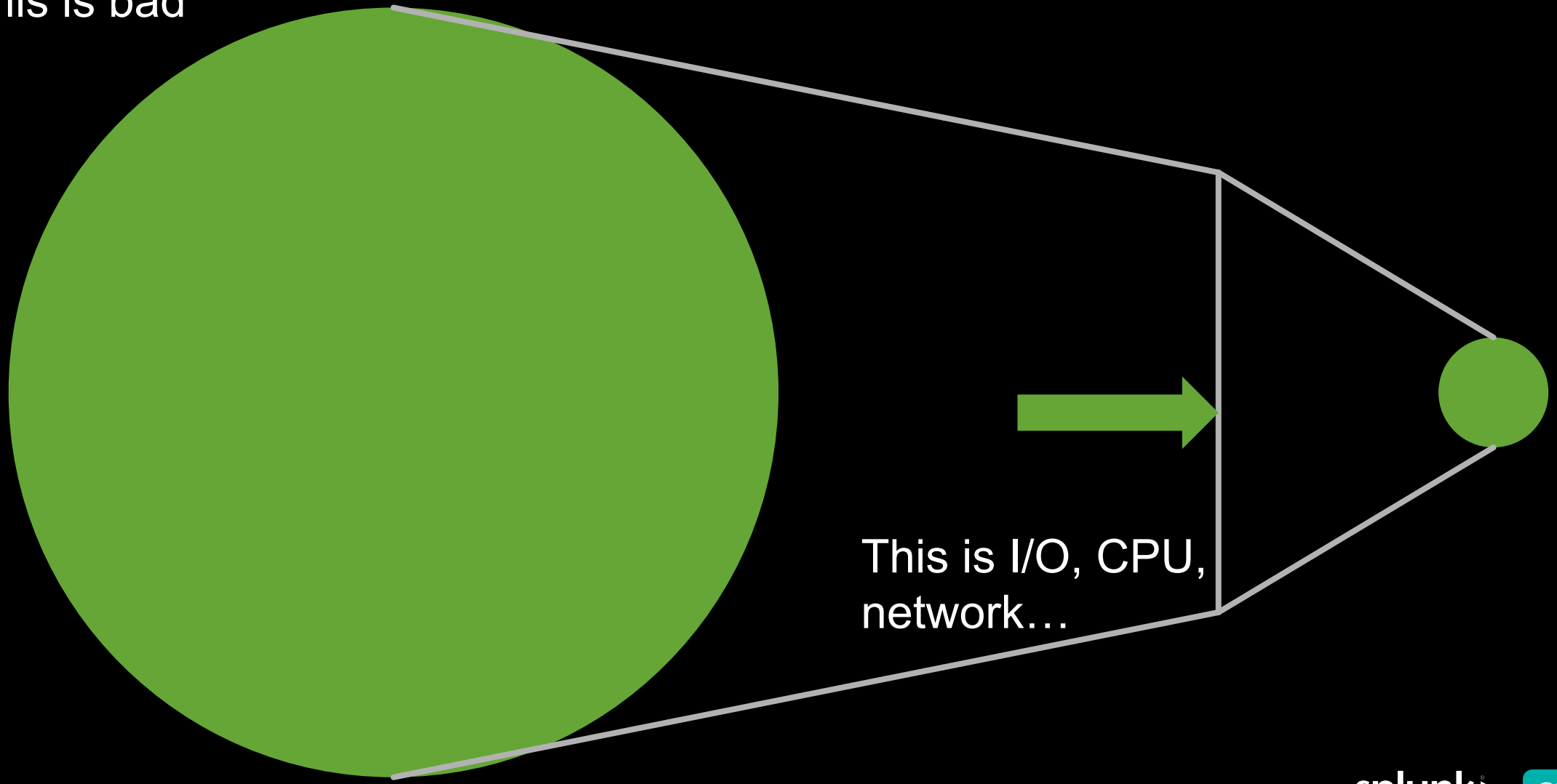


The Search Funnel



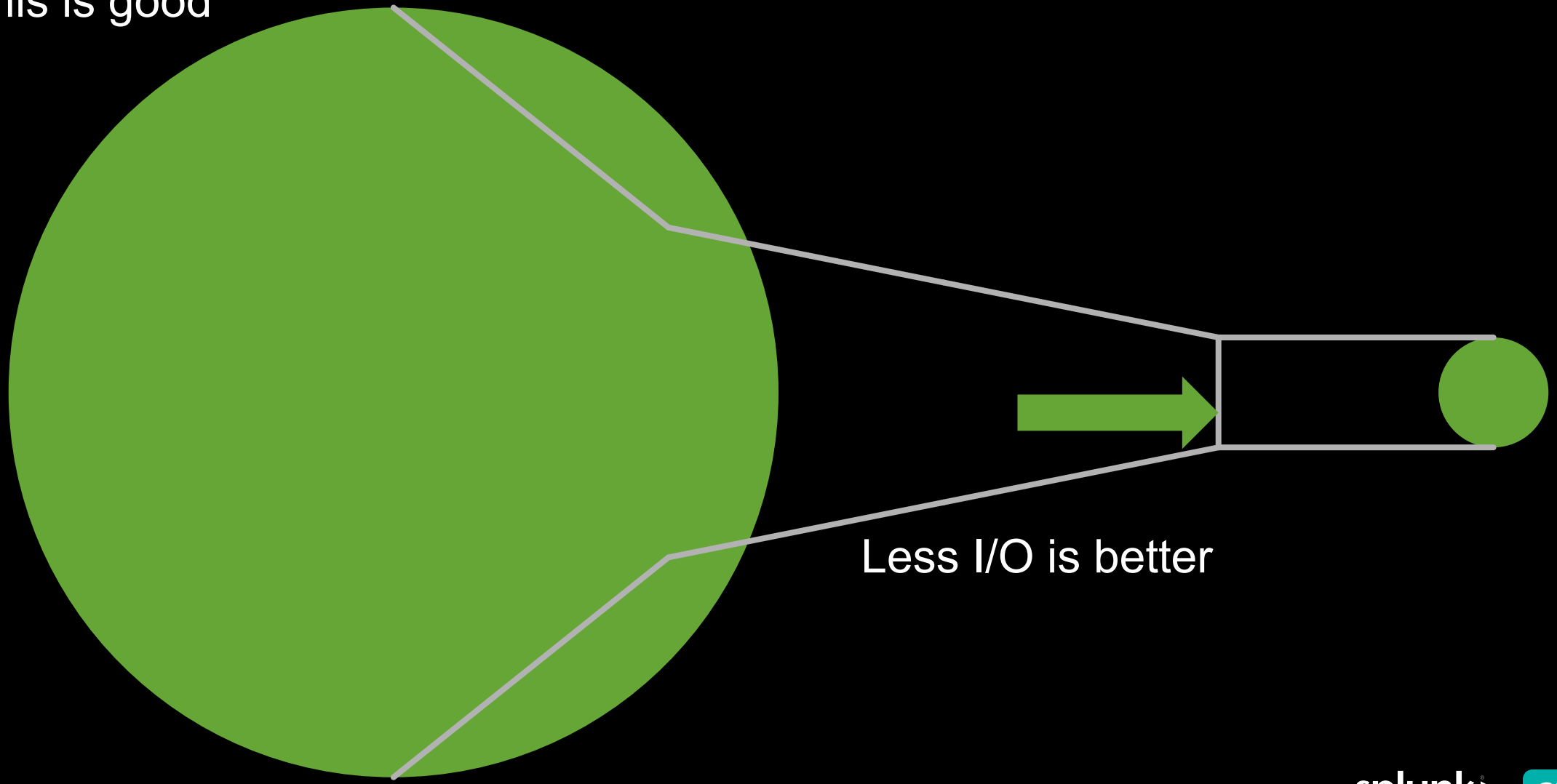
The Search Funnel

- This is bad



The Search Funnel

- This is good



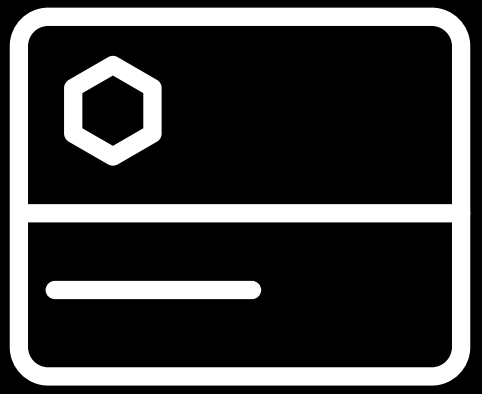
A Brief Overview Of Splunk

Splunk 101

splunk> .conf2017

A Brief Overview Of Splunk

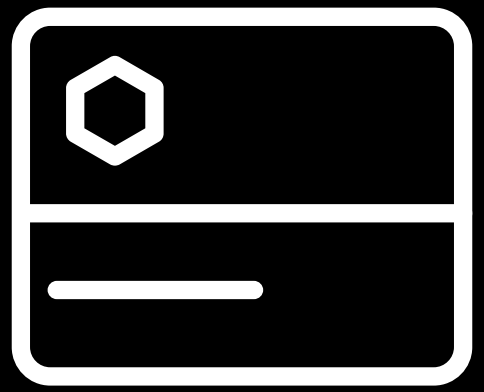
This is an indexer



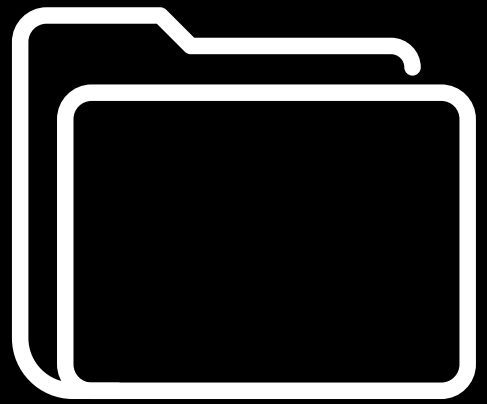
Indexer

A Brief Overview Of Splunk

Indexers contain indexes



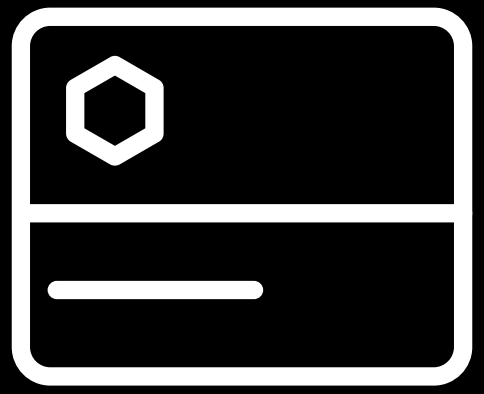
Indexer



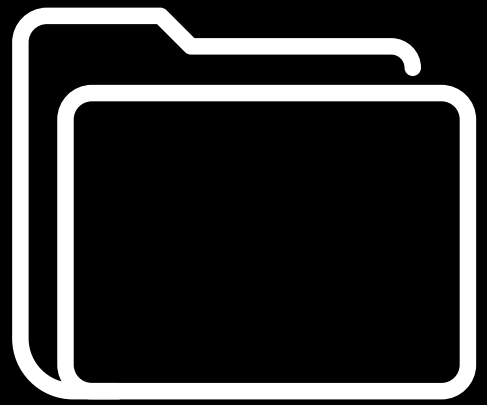
Index

A Brief Overview Of Splunk

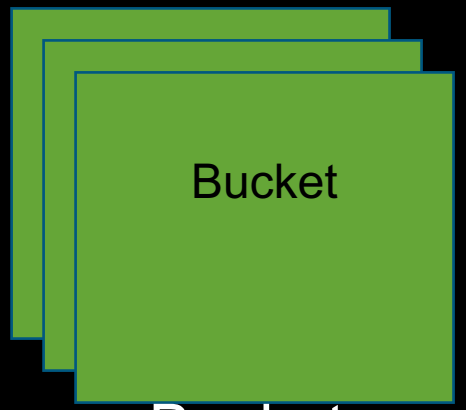
Indexes contain buckets



Indexer



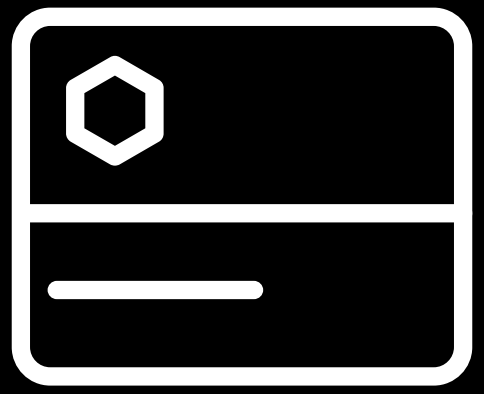
Index



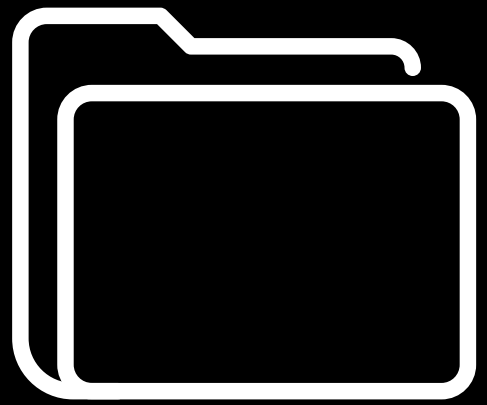
Buckets

A Brief Overview Of Splunk

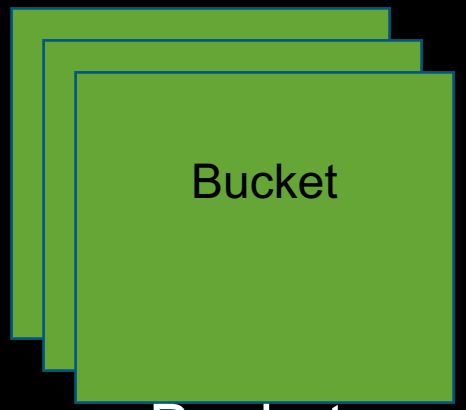
Buckets contain logs



Indexer



Index

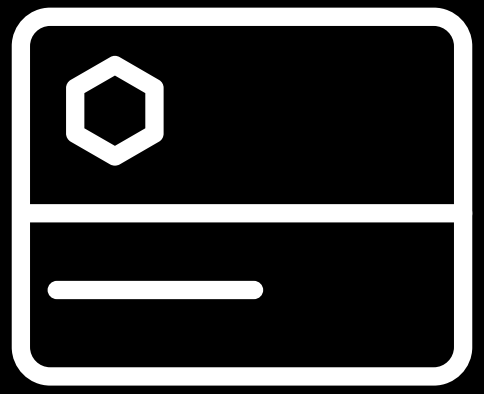


Buckets

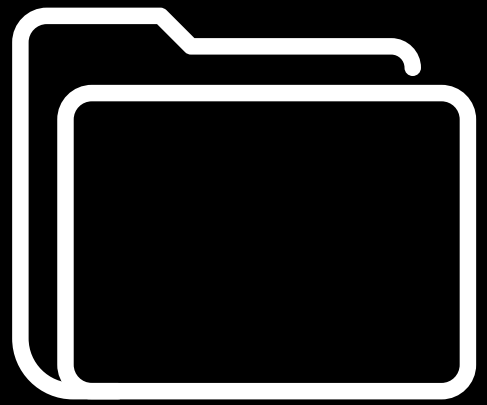
Logs

A Brief Overview Of Splunk

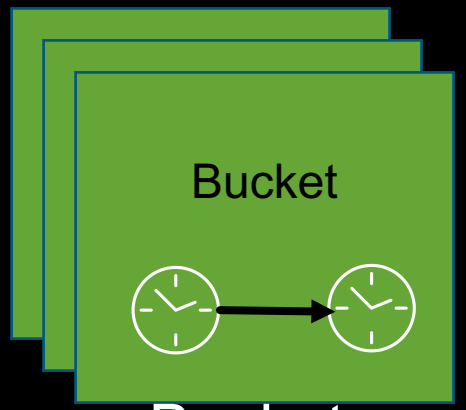
… and a timestamp range



Indexer



Index

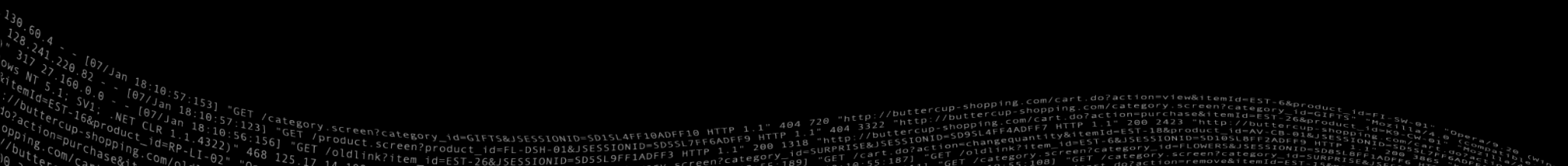


Buckets
+time

Logs

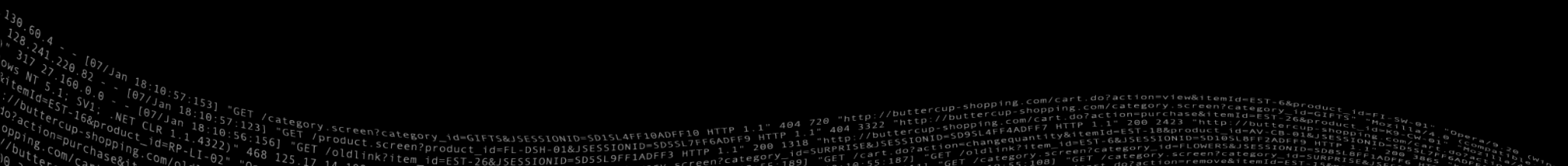
The Easy Stuff

- Did the search ask for an index to be searched?



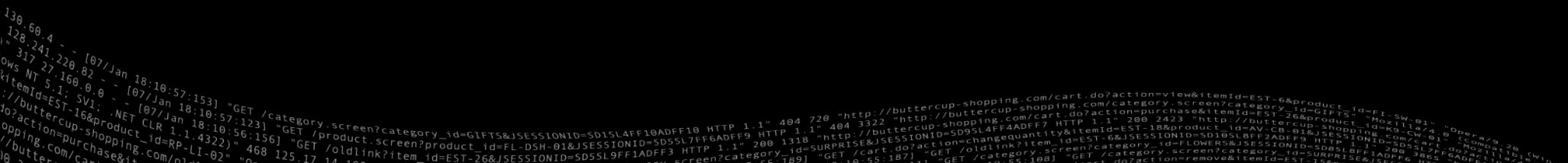
The Easy Stuff

- Did the search ask for an index to be searched?
  - Ignore the index!



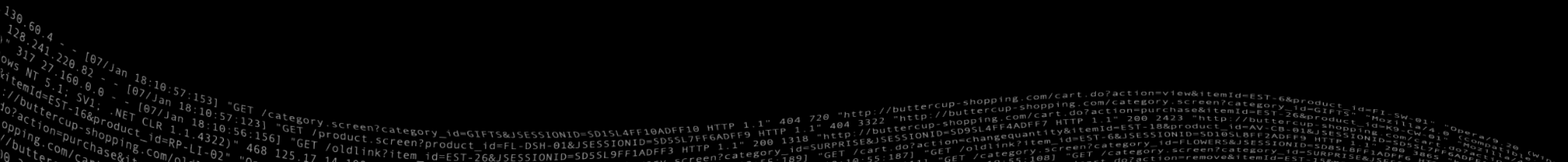
The Easy Stuff

- Did the search ask for an index to be searched?
  - Ignore the index!
- Does the search even have permission to search the index?



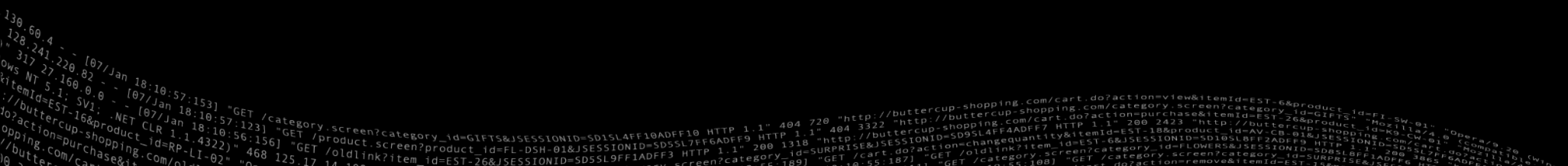
The Easy Stuff

- Did the search ask for an index to be searched?
  - Ignore the index!
- Does the search even have permission to search the index?
  - Ignore the index!



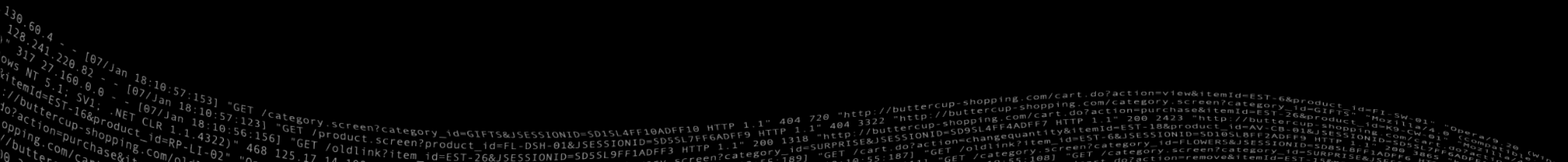
The Easy Stuff

- Did the search ask for an index to be searched?
  - Ignore the index!
- Does the search even have permission to search the index?
  - Ignore the index!
- Does this bucket time range overlap the search time range?



The Easy Stuff

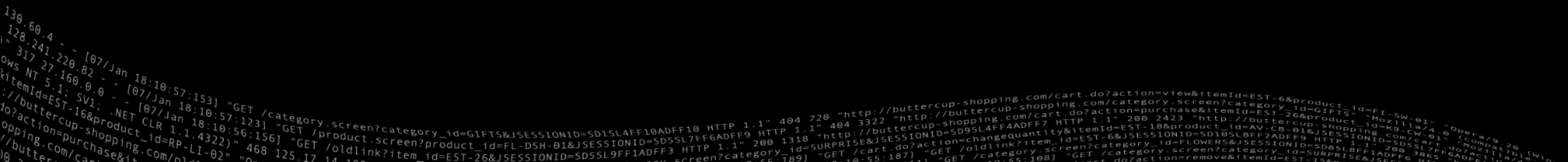
- Did the search ask for an index to be searched?
  - Ignore the index!
- Does the search even have permission to search the index?
  - Ignore the index!
- Does this bucket time range overlap the search time range?
  - Ignore the bucket!



The Easy Stuff

- Did the search ask for an index to be searched?
  - Ignore the index!
- Does the search even have permission to search the index?
  - Ignore the index!
- Does this bucket time range overlap the search time range?
  - Ignore the bucket!

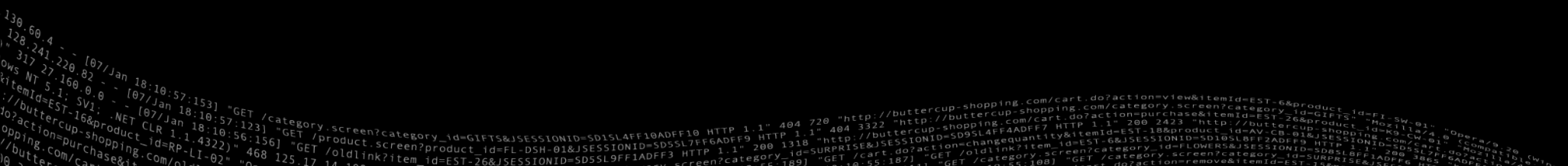
Being lazy is awesome! The more we can ignore, the less work we do.



What's In A Bucket?

Timeseries index files

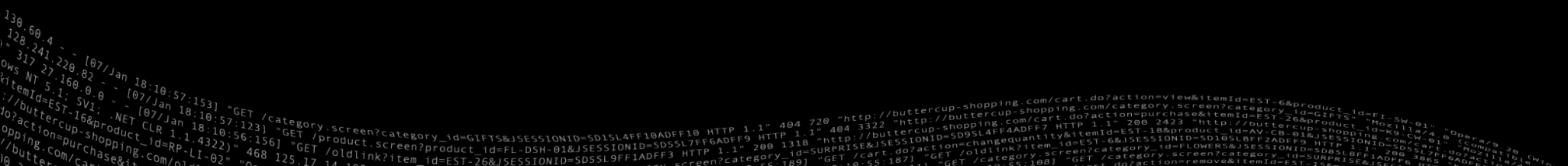
```
-rw------- 1 splunk splunk 10500393 Jul 31 06:01 1501478535-1496207100-0863.tsidx
-rw------- 1 splunk splunk  4804861 Jul 31 07:05 1501479625-1501478489-8260.tsidx
```



What's In A Bucket?

The Bloom filter

```
-rw------- 1 splunk splunk 10500393 Jul 31 06:01 1501478535-1496207100-0863.tsidx
-rw------- 1 splunk splunk  4804861 Jul 31 07:05 1501479625-1501478489-8260.tsidx
-rw------- 1 splunk splunk   424909 Jul 31 07:06 bloomfilter
```



splunk> .conf2017

What's In A Bucket?

Metadata!

```
-rw------- 1 splunk splunk 10500393 Jul 31 06:01 1501478535-1496207100-0863.tsidx
-rw------- 1 splunk splunk  4804861 Jul 31 07:05 1501479625-1501478489-8260.tsidx
-rw------- 1 splunk splunk   424909 Jul 31 07:06 bloomfilter
-rw------- 1 splunk splunk       75 Jul 31 07:06 bucket_info.csv
-rw------- 1 splunk splunk    15462 Jul 31 07:06 Hosts.data
-rw------- 1 splunk splunk  3713177 Jul 31 07:06 merged_lexicon.lex
-rw------- 1 splunk splunk       49 Jul 31 07:06 optimize.result

-rw------- 1 splunk splunk     6875 Jul 31 07:06 Sources.data
-rw------- 1 splunk splunk     7024 Jul 31 07:06 SourceTypes.data
-rw------- 1 splunk splunk       77 Jul 31 07:06 splunk-autogen-params.dat
```

splunk> .conf2017

What's In A Bucket?

All your logs (they're compressed)

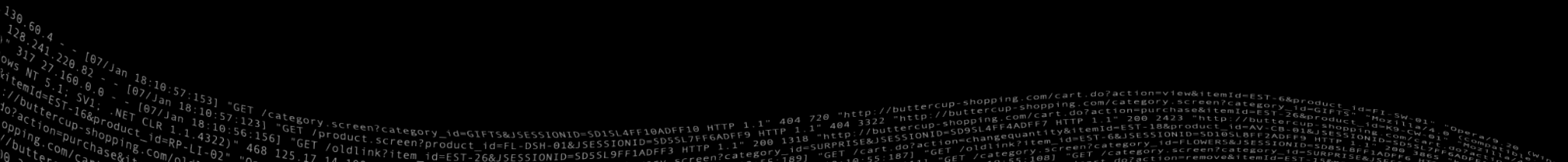
```
-rw------- 1 splunk splunk 10500393 Jul 31 06:01 1501478535-1496207100-0863.tsidx
-rw------- 1 splunk splunk  4804861 Jul 31 07:05 1501479625-1501478489-8260.tsidx
-rw------- 1 splunk splunk   424909 Jul 31 07:06 bloomfilter
-rw------- 1 splunk splunk       75 Jul 31 07:06 bucket_info.csv
-rw------- 1 splunk splunk    15462 Jul 31 07:06 Hosts.data
-rw------- 1 splunk splunk  3713177 Jul 31 07:06 merged_lexicon.lex
-rw------- 1 splunk splunk       49 Jul 31 07:06 optimize.result
drwx------ 2 splunk splunk     4096 Jul 31 07:06 rawdata
-rw------- 1 splunk splunk     6875 Jul 31 07:06 Sources.data
-rw------- 1 splunk splunk     7024 Jul 31 07:06 SourceTypes.data
-rw------- 1 splunk splunk       77 Jul 31 07:06 splunk-autogen-params.dat
```

splunk> .conf2017

Bloom Filters

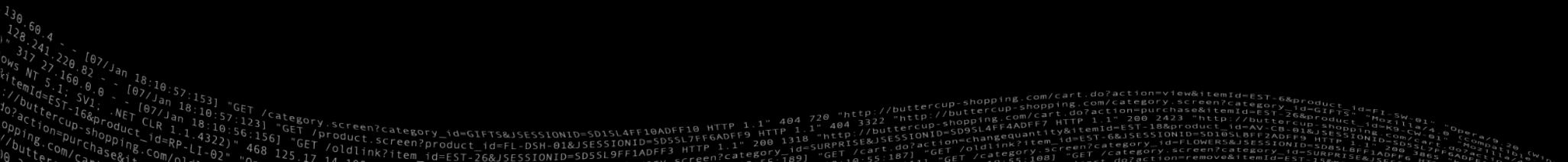
Bloom Filters

- Probabilistic data structure
  - They tell you if an item might be present, or if an item is not present

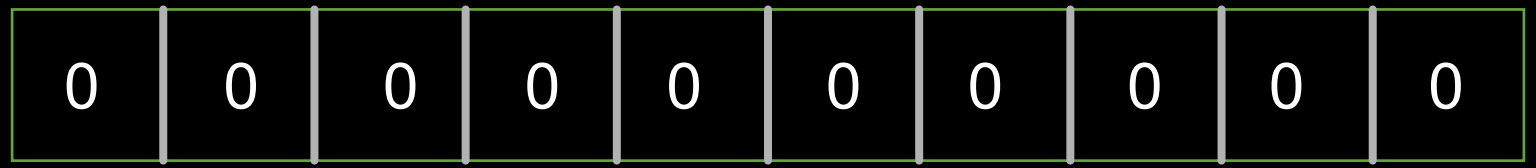


Bloom Filters

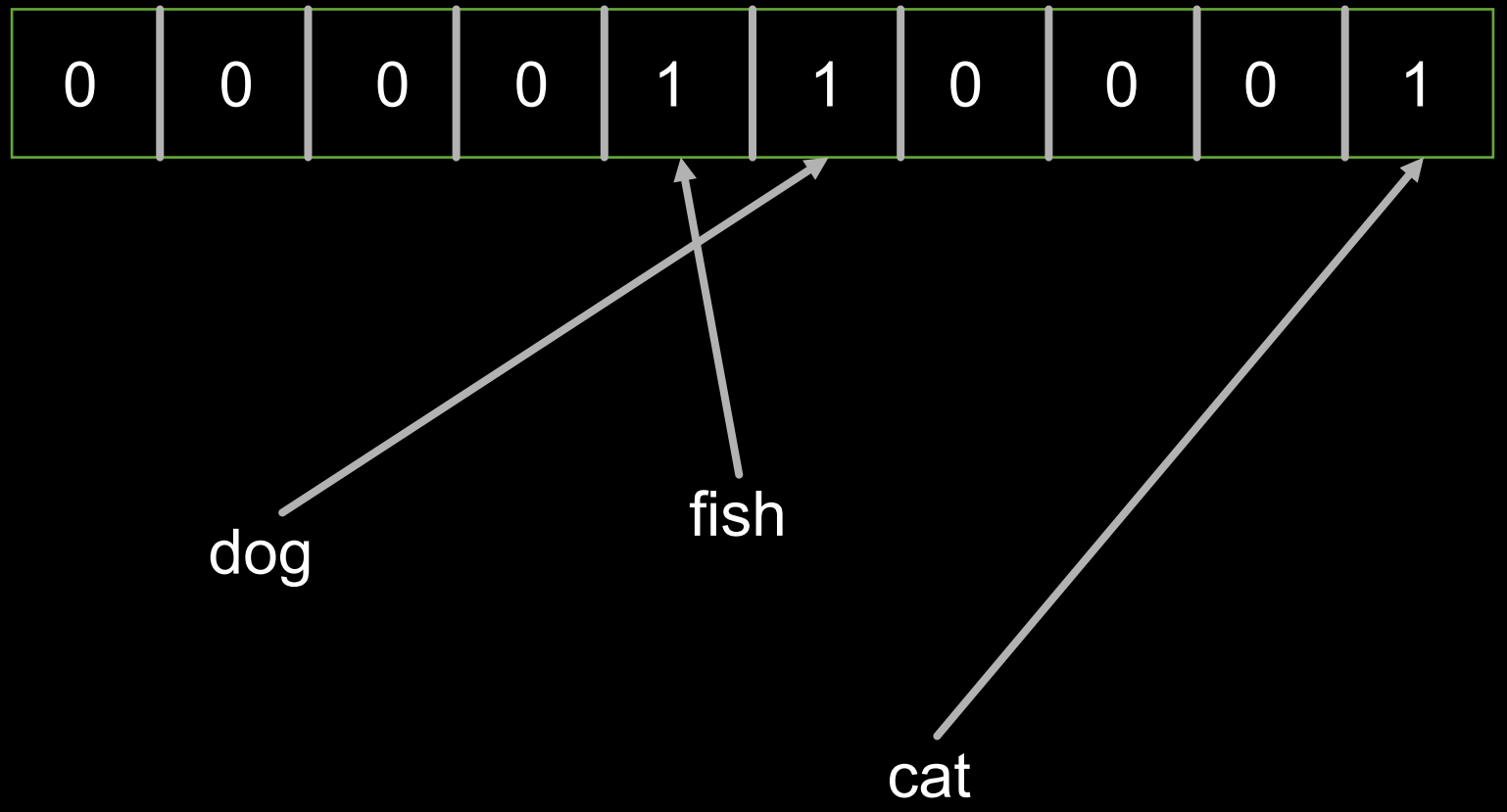
- Probabilistic data structure
  - They tell you if an item might be present, or if an item is not present
- They make a space / accuracy trade off



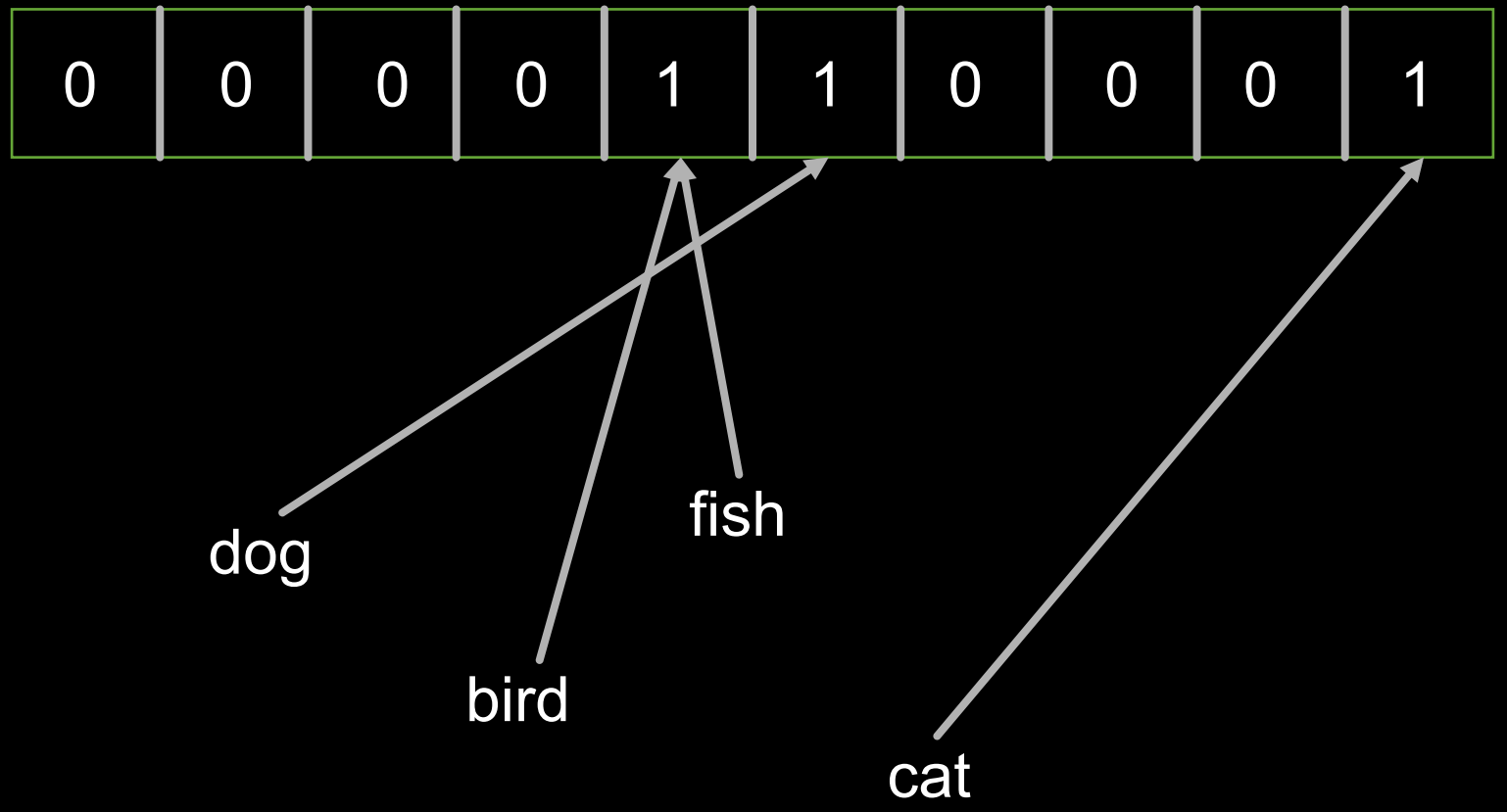
Bloom Filters



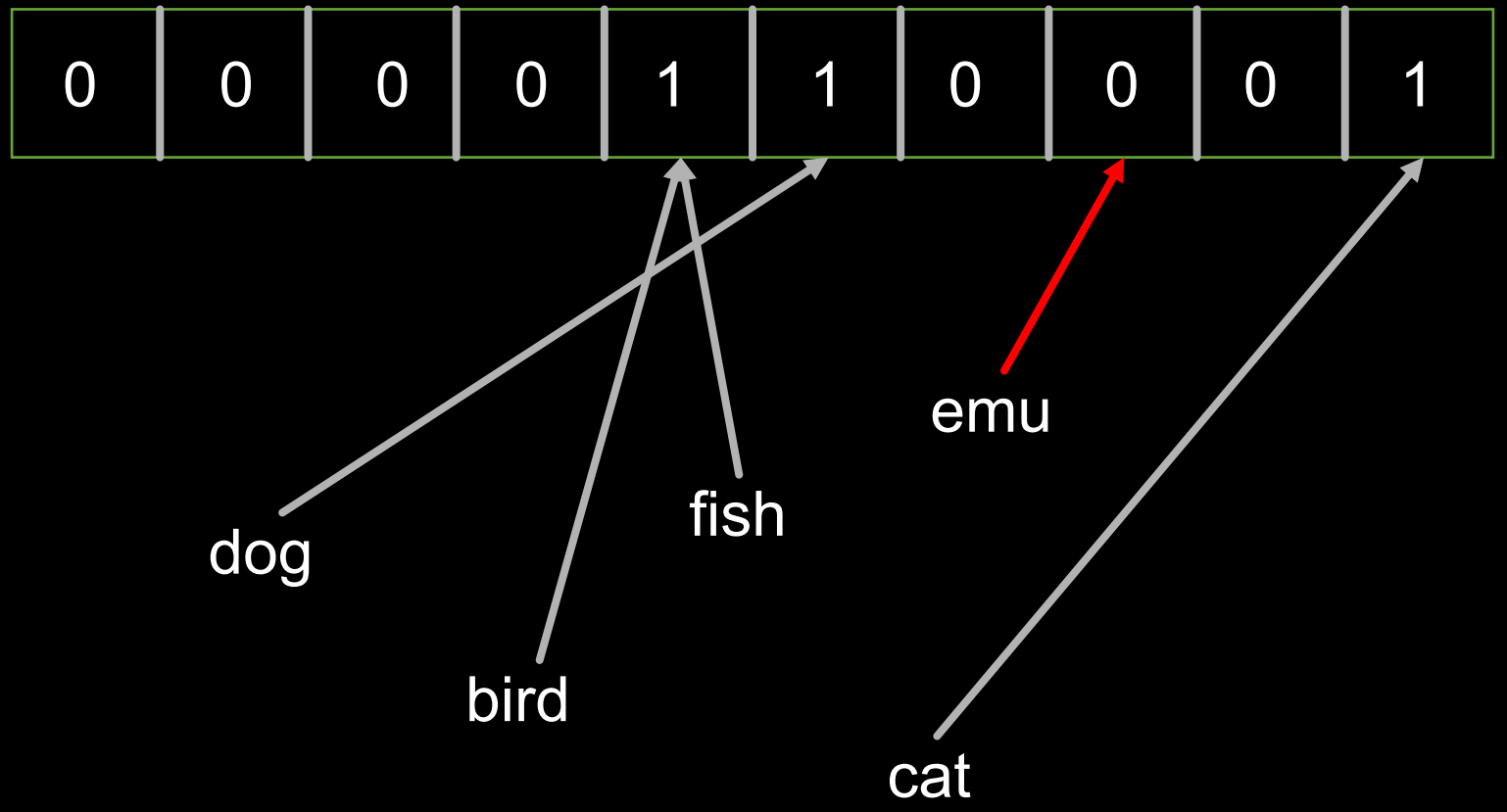
Bloom Filters



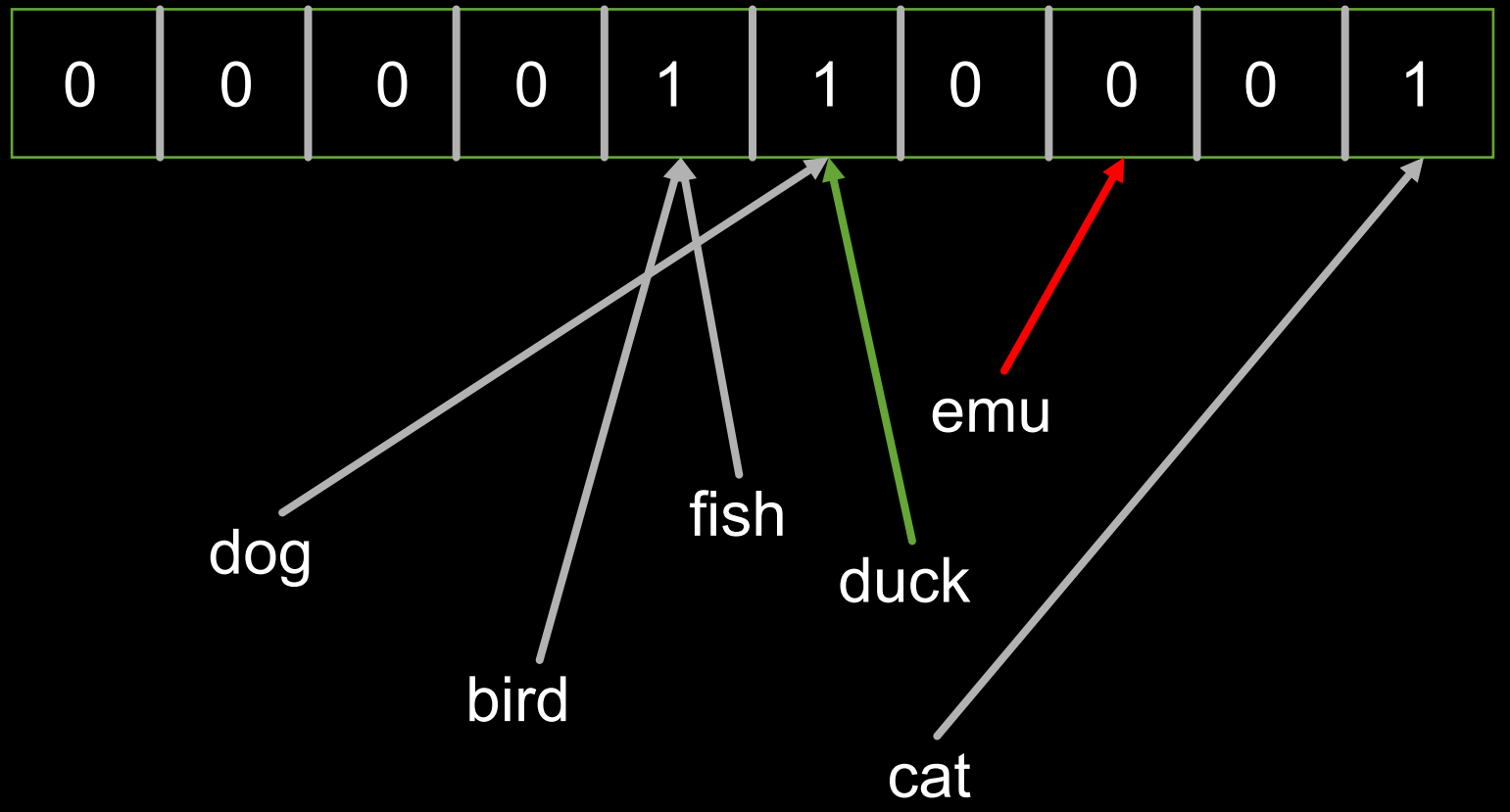
Bloom Filters



Bloom Filters

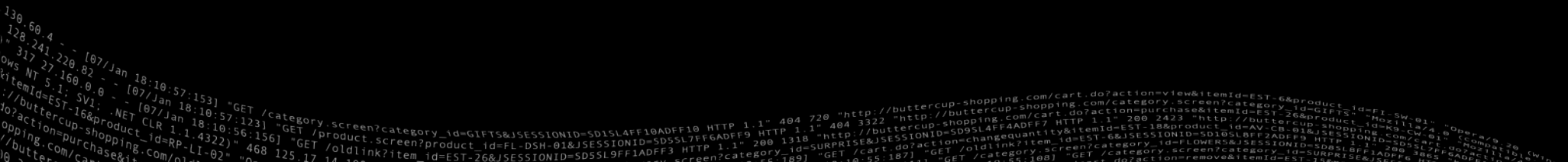


Bloom Filters



Bloom Filters

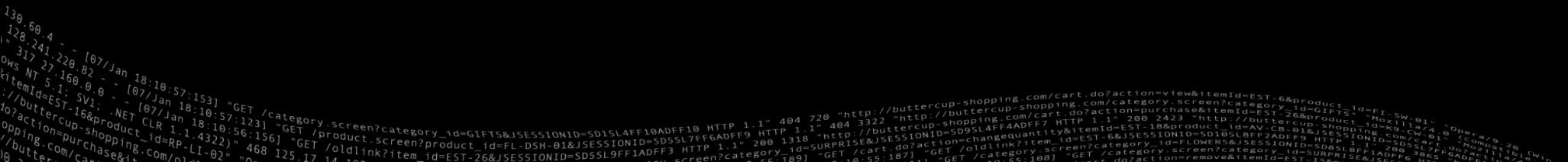
Let’s write some code



splunk> .conf2017

Bloom Filters

They probably work.



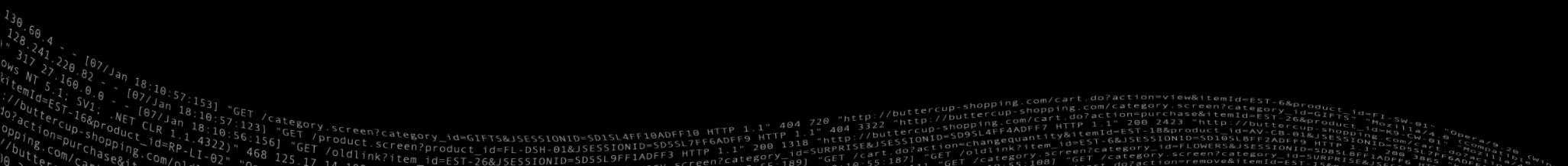
splunk> .conf2017

Terms And Segmenting

- Major segmenting

Terms And Segmenting

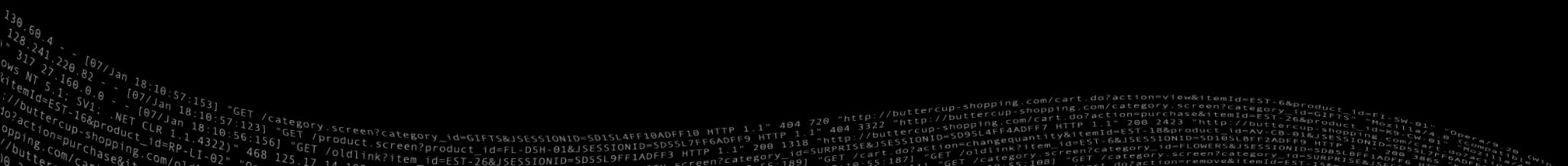
- Major segmenting
  - Split event by a set of “major breaks”
  - Configured in segmenters.conf
    - [ ] < > ( ) { } | ! ; , ' " * \n \r \s \t & ? + %21 %26 %2526 %3B %7C %20 %2B %3D -- %2520 %5D %5B %3A %0A %2C %28 %29



splunk> .conf2017

Terms And Segmenting

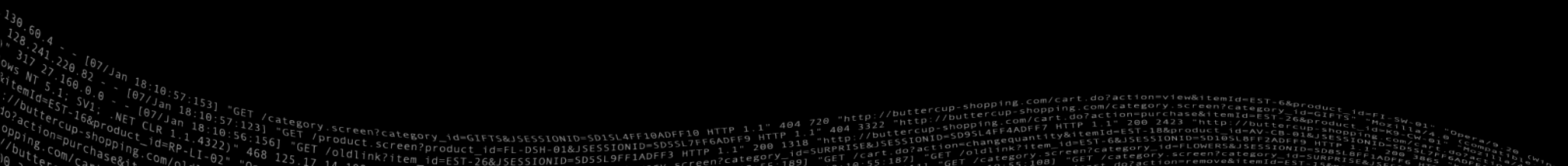
- Major segmenting
  - Split event by a set of “major breaks”
  - Configured in segmenters.conf
    - [ ] < > ( ) { } | ! ; , ' " * \n \r \s \t & ? + %21 %26 %2526 %3B %7C %20 %2B %3D -- %2520 %5D %5B %3A %0A %2C %28 %29
  - We’re lazy!
    - Single characters only!
    - “ \s



Terms And Segmenting

- Major segmenting
  - Split event by a set of “major breaks”
  - Configured in segmenters.conf
    - [ ] < > ( ) { } | ! ; , ' " * \n \r \s \t & ? + %21 %26 %2526 %3B %7C %20 %2B %3D -- %2520 %5D %5B %3A %0A %2C %28 %29
  - We’re lazy!
    - Single characters only!
    - “ \s

src_ip = 1.2.3.4

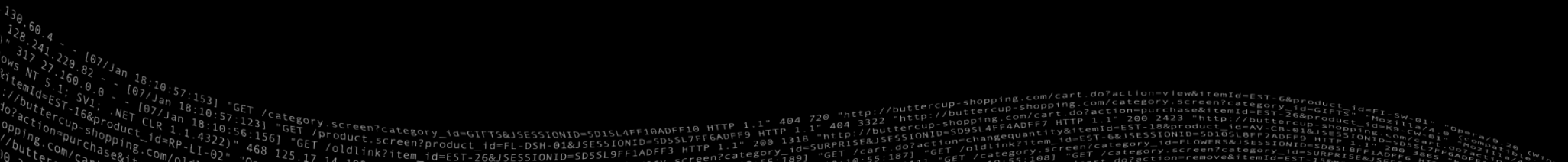


splunk> .conf2017

Terms And Segmenting

- Major segmenting
  - Split event by a set of “major breaks”
  - Configured in segmenters.conf
    - [ ] < > ( ) { } | ! ; , ' " * \n \r \s \t & ? + %21 %26 %2526 %3B %7C %20 %2B %3D -- %2520 %5D %5B %3A %0A %2C %28 %29
  - We’re lazy!
    - Single characters only!
    - “ \s

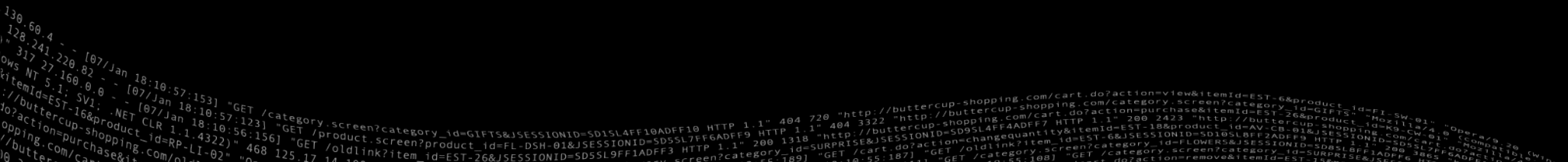
src_ip = 1.2.3.4



splunk> .conf2017

Terms And Segmenting

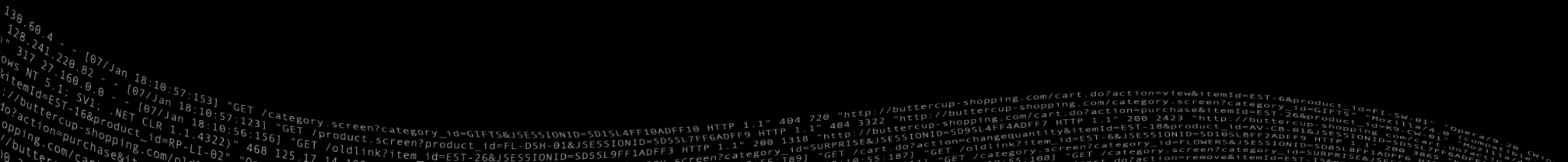
- Minor segmenting



splunk> .conf2017

Terms And Segmenting

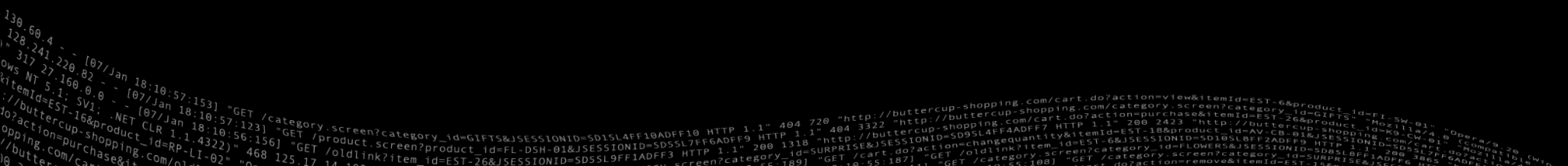
- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _



splunk> .conf2017

Terms And Segmenting

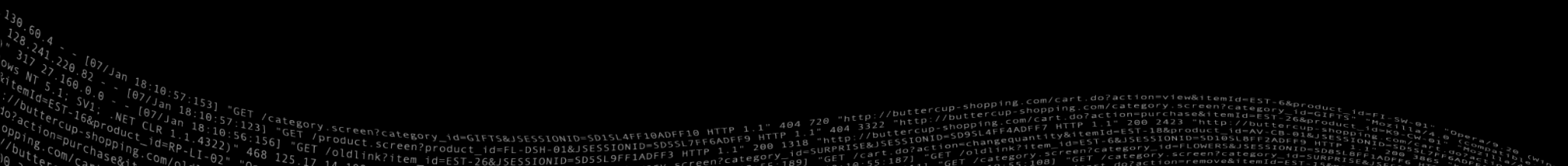
- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character



Terms And Segmenting

- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character

1.2.3.4



Terms And Segmenting

- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character

1.2.3.4

Terms And Segmenting

- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character

1.2.3.4

Terms And Segmenting

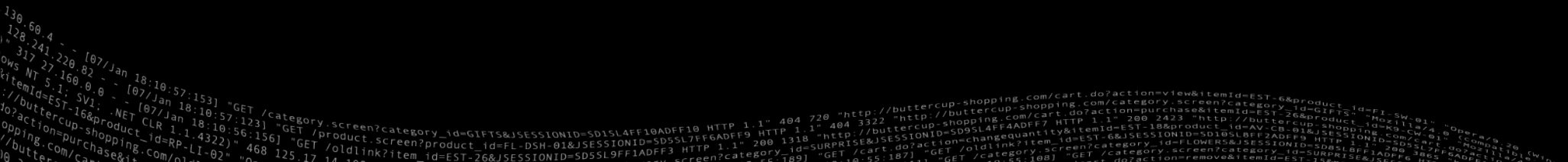
- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character

1.2.3.4

Terms And Segmenting

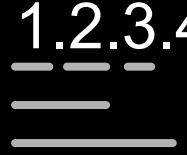
- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character

1.2.3.4



Terms And Segmenting

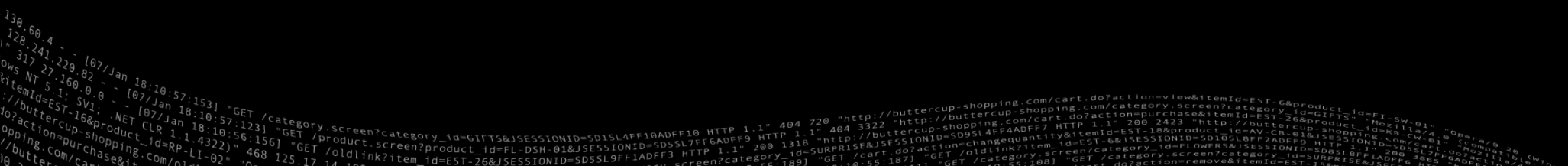
- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character



Terms And Segmenting

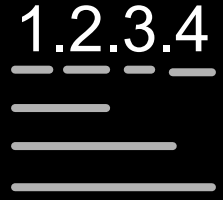
- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character

1.2.3.4



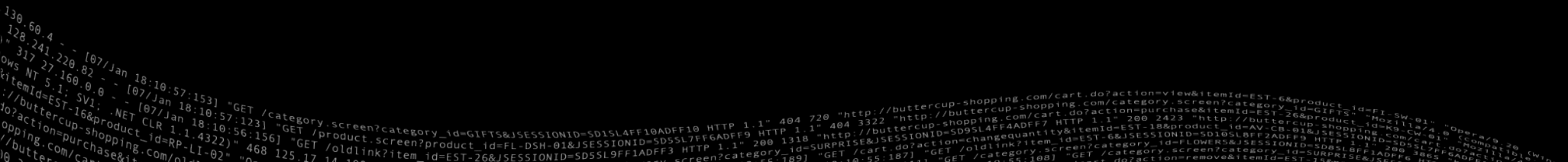
Terms And Segmenting

- Minor segmenting
  - Split a major segment by a set of “minor breaks”
  - Configured in segmenters.conf
    - / : = @ . - $ # % \ _
  - Track multiple minor segments
    - From the end of the last minor break to the next minor break
    - From the start of the major segment to the minor break character



The Timeseries Index

- Every event is given an ID unique within its bucket
- The event is segmented in to terms
- Each term is tracked in the TSIDX
- Each term is associated with a list of events



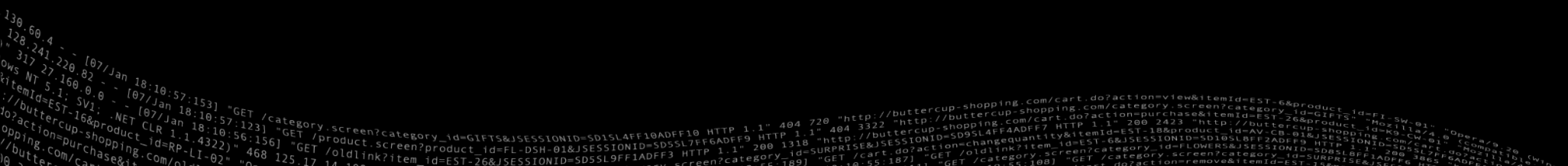
splunk> .conf2017

The Timeseries Index

- Every event is given an ID unique within its bucket
- The event is segmented in to terms
- Each term is tracked in the TSIDX
- Each term is associated with a list of events

deployed	(1)
east	(1)
in	(1)
service	(1)

1. service deployed in east



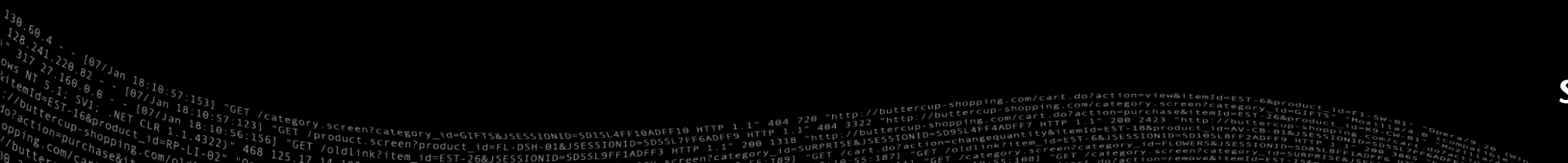
splunk> .conf2017

The Timeseries Index

- Every event is given an ID unique within its bucket
- The event is segmented in to terms
- Each term is tracked in the TSIDX
- Each term is associated with a list of events

deployed	(1, 2)
east	(1)
in	(1, 2)
service	(1, 2)
west	(2)

1. service deployed in east
2. service deployed in west

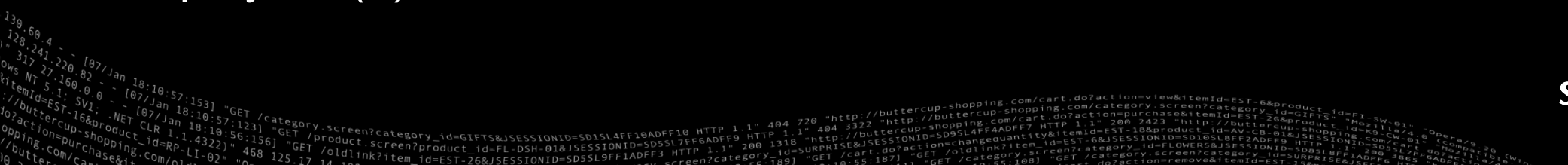


The Timeseries Index

- Every event is given an ID unique within its bucket
- The event is segmented in to terms
- Each term is tracked in the TSIDX
- Each term is associated with a list of events

deployed	(1, 2)
east	(1, 3)
in	(1, 2, 3)
service	(1, 2, 3)
west	(2)
undeployed	(3)

1. service deployed in east
2. service deployed in west
3. service undeployed in east



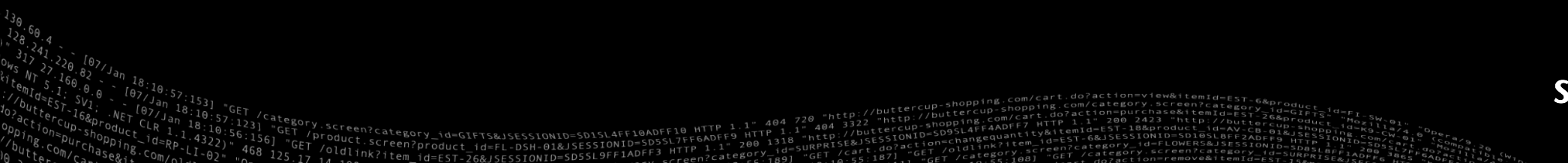
The Timeseries Index

splunk> .conf2017

The Timeseries Index

Stuff not implemented

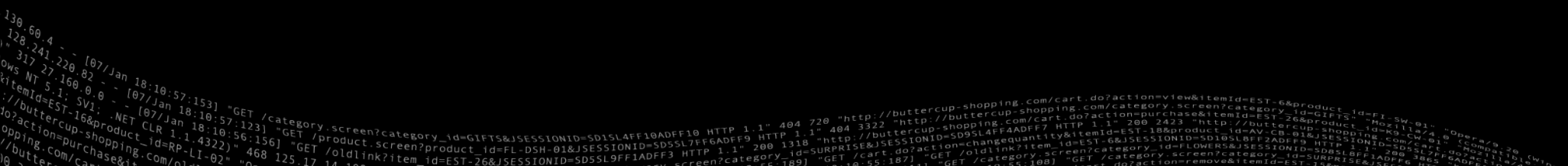
- The list of terms is stored in lexographical order
- This list is compressed in blocks of N terms
- The blocks can be searched using a basic binary search to find if a term is potentially in a block
- If a term is potentially in the block, the block can be scanned linearly to confirm



The Timeseries Index

A note on wildcards

- It is possible to efficiently search with wildcards if they don't start with a *:
- It is possible to inefficiently search for wildcards that do start with a *
  - Decompressing and scanning all terms is still faster than decompressing and scanning all logs
- Wildcards bypass the Bloom filter
- There is probably some more secret sauce in this area since I did my research



Takeaways

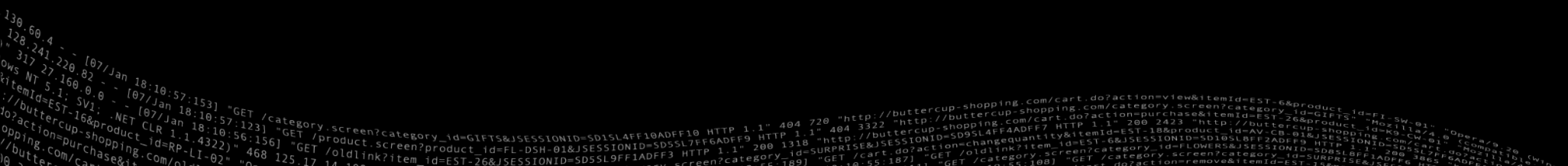
For log consumers

- When searching, the more you can put before the |, the better
- Generic terms like “error” or “warning” are not great (on their own)
- Multiple generic terms (“an error has occurred”) are relatively better
- Specific terms are good, but may not be available
- Wildcards work better if combined with terms, because they bypass the bloom filter
- “Add to search” can sometimes create an inefficient search (“| spath | search”)

Takeaways

For log producers

- Emit specific and unique terms for specific things
  - Error=13, Warning=21 are bad (error, warning, 13, and 21 are very generic)
  - Error13, Warning21 are better
  - ErrorUserNotFound, InfoPurchaseStatus are good (human parseable)
- Lots of common terms are bad
  - “system status is up”, “cart step completed”
  - The set of filtered events takes longer to calculate, and it may still be much larger than it should
- Better
  - “SystemStatusUp”, “CartStepCompleted”
  - These are very specific terms. They won’t occur randomly.

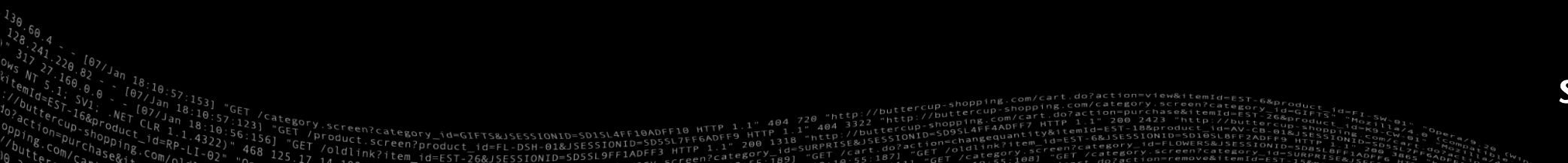


splunk> .conf2017

Takeaways

For both

- Be selective with minor breaks. They can substitute for wildcards but more terms also has a cost. Sample terms: Cart_Created, Cart_NextStep, Cart_Lost
  - “Cart” will match all 3 using the bloom filter and index
  - “Cart*” will search for “Cart*” using the index (no bloom filter)
  - “Cart_*” will search for “Cart” using the bloom filter and index, read+decompress logs, then look for “Cart_”
  - “Cart_Created” will search for “Cart AND Created”, read+decompress logs, then search those for “Cart_Created”
  - “TERM(Cart_Created)” will search for “Cart_Created” using the bloom filter and index

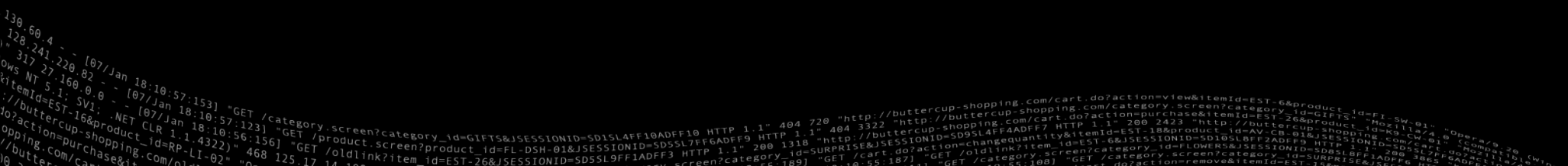


splunk> .conf2017

Takeaways

Look at your lispy

- Lispy is the language that drives the filter which decides what to pull from disk
- Access it via “Inspect Job”, selecting “search.log”, and then searching the information dump for “lispy”



splunk> .conf2017

Takeaways

Lispy examples

- 08-14-2017 10:55:37.069 INFO UnifiedSearch - Expanded index search = cart_*
- 08-14-2017 10:55:37.069 INFO UnifiedSearch - base lispy: [AND cart]

…

- 08-14-2017 10:59:56.899 INFO UnifiedSearch - Expanded index search = cart*
- 08-14-2017 10:59:56.899 INFO UnifiedSearch - base lispy: [AND cart*]

…

- 08-14-2017 11:00:32.346 INFO UnifiedSearch - Expanded index search = cart_created
- 08-14-2017 11:00:32.346 INFO UnifiedSearch - base lispy: [AND created cart]

…

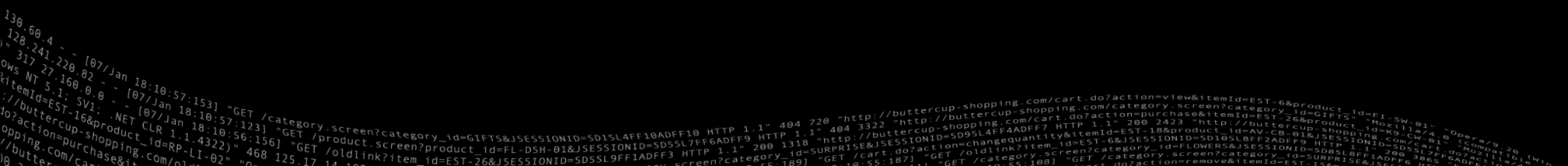
- 08-14-2017 11:01:58.559 INFO UnifiedSearch - Expanded index search = TERM(cart_created)
- 08-14-2017 11:01:58.559 INFO UnifiedSearch - base lispy: [AND cart_created]

A Final Warning

splunk> .conf2017

A Final Warning

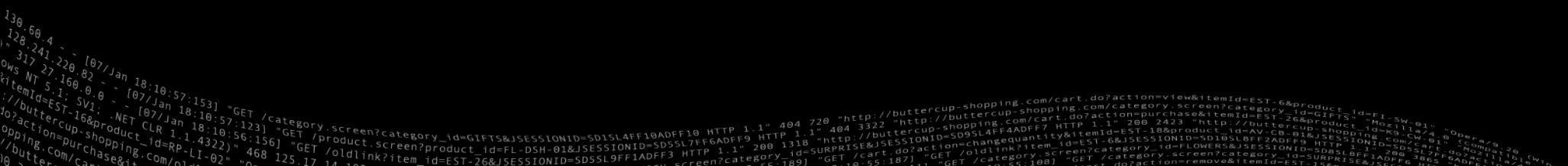
- If you go too far, you may find:
  - Trading “human” data for “machine” data may reduce your license consumption



splunk> .conf2017

A Final Warning

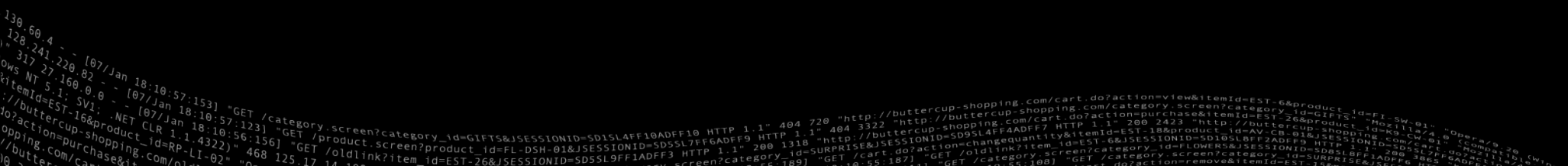
- If you go too far, you may find:
  - Trading “human” data for “machine” data may reduce your license consumption
  - Less I/O may reduce your hardware requirements



splunk> .conf2017

A Final Warning

- If you go too far, you may find:
  - Trading “human” data for “machine” data may reduce your license consumption
  - Less I/O may reduce your hardware requirements
  - Faster searches may mean less sword fighting, more analyzing logs



Thank You

Don't forget to rate this session in the .conf2017 mobile app