Data Integrity Control

Dhruva Bhagi
Sr Software Engineer, Splunk

splunk>

Disclaimer

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 the 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.

Problem Statement

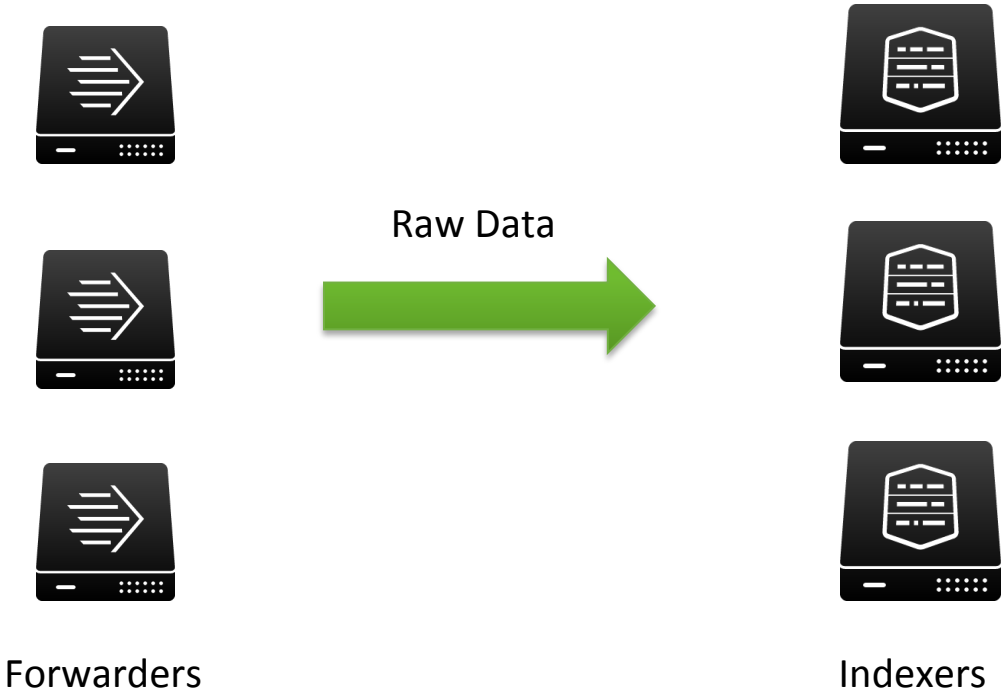
- As Splunk moves up higher in value chain and becomes a single source of truth for all machine data, Its important to provide a mechanism to verify & safeguard the integrity of the indexed data

- Typical use cases are:
  - To be able to prove the integrity of the data for auditing & legal purposes
  - To be able to check the integrity of the indexed raw-data on demand, say after any unforeseen activities like intrusion etc.

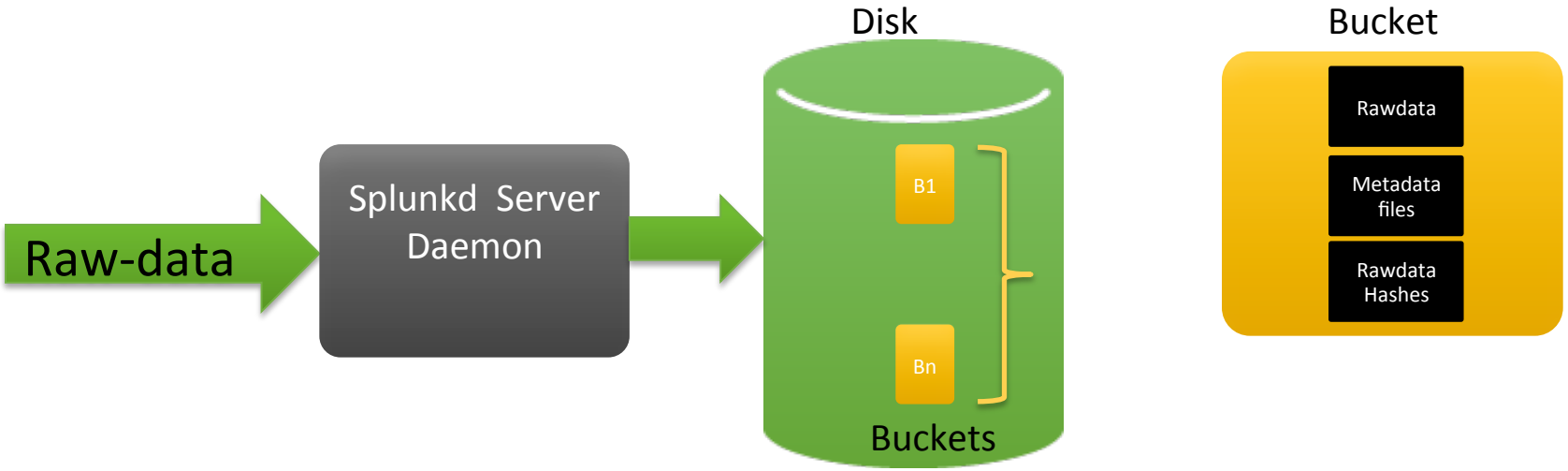
Goals

- Provide a mechanism to check the integrity of indexed raw-data
- Build a basic acceptable framework into the product so that customers can later configure/extend it as per their needs
- Make it work in a distributed environments (with indexer clustering)
- Keep it simple & easy to understand

Topology



At Indexer


Raw-data
Splunkd Server Daemon
Disk
B1
Bn
Buckets
Bucket
Rawdata
Metadata files
Rawdata Hashes

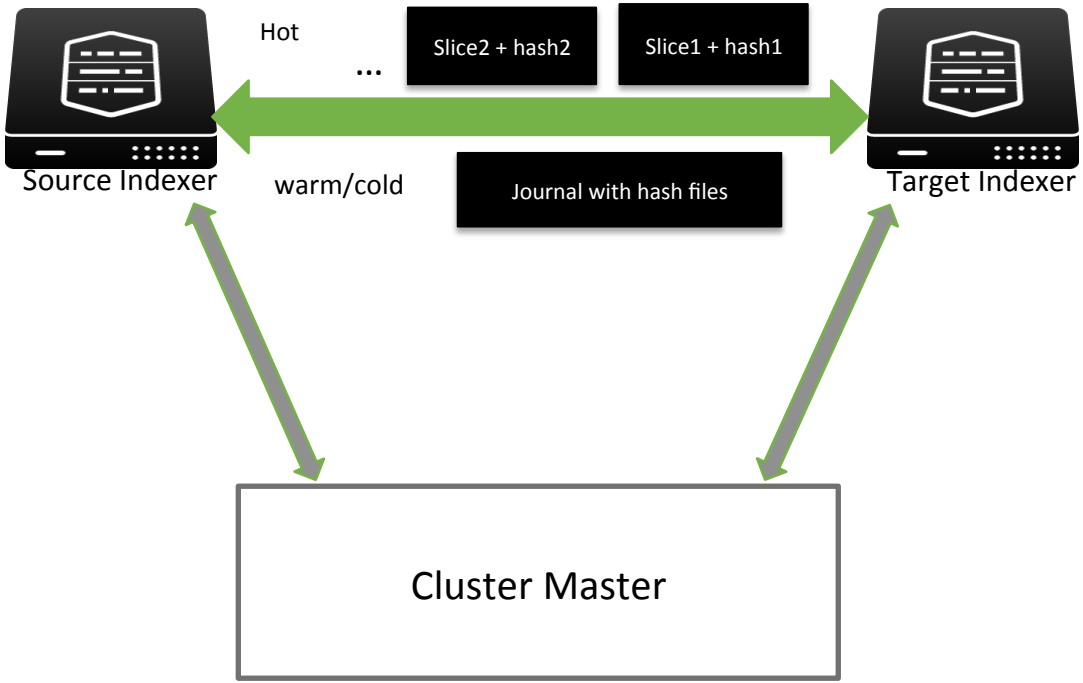
How it Works

- Splunk buckets are comprised of a unit of data called “Slice”
- Each slice is compressed & written to the journal as it comes
- When you turn on this feature, We compute a SHA256 hash on each compressed and write it to the disk (journal & external hash file called l1Hashes_BucketId_guid.dat.tmp) as it comes
- When bucket gets rolled from hot to warm, we finalize hash files:
  - We compute a hash of all hashes in l1Hashes & write it to l2Hash file
  - We rename l1Hashes_bucketID_guid.dat.tmp to l1Hashes_bucketID_guid.dat
- From this point, we don’t write to the bucket/hash files anymore

In Indexer Cluster

- In case of indexer clustering, Hot replication happens slice by slice
- If this feature is turned on, we replicate (slice + hash) as one data unit & the targets will start building up the hash files just like source
- In the case of warm/cold replications, we just replicate hash files along with the journal for the Data Integrity control enabled buckets
- In short, underlying framework takes care of hash replication

Cluster Topology



How to Enable/Disable

- indexes.conf Has a new global setting “enableDataIntegrityControl” which defaults to false
- So, this feature is disabled by default for all indexes
- Below is how you turn on this feature for a given index, say index1:
  [index1]
  enableDataIntegrityControl=true
- In indexer cluster environment, the recommended way to turn ON this feature is by pushing a bundle from cluster master (using apply cluster-bundle command)
- This setting doesn’t require a restart (reloadable)

Checking Integrity

- If you want to check the integrity of a bucket/index at an Indexer, just run below CLI:
  - splunk check-integrity –bucketPath <path_to_bucket>
  - splunk check-integrity –index <index_name>
- Integrity check happens in a separate process so that it won't create any direct impact on splunkd process
- The way this works is:
  - We first validate the l1Hashes file against l2Hash file
  - If the above step succeeds, we go ahead & validate rawdata slice by slice
- On integrity check failures, splunk provides bucket & the slice number which got tampered with

Performance Metrics

Feature ON/OFF	Dataset	Time to index	Time to verify	Notes
OFF	50 GB	38.18 Mins		Local indexing
ON	50 GB	39.03 Mins	21.25 Mins	Local Indexing
OFF	200 GB	169.5 Mins	-	4 UF->Indexer
ON	200 GB	174 Mins	112.39	4 UF->Indexer
OFF	400 GB	344.72	-	8 UF->1 Indexer
ON	400 GB	352.96	235.46	8 UF->1 Indexer

- Based on the above numbers from our performance team, there is no much CPU overhead when we index data with Data Integrity Control enabled
- These numbers are generated with a max throughput rate of ~25MB per sec that a splunk indexer can handle

Hardware Used

Below is the hardware configuration on which performance testing was done:

- HP DL380G8
- Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz
  - 32cores with Hyper Threading
- 5 x 146Gb SAS 15k RPM raids in RAID-10
- 64GB RAM

Disk Overhead

- For Every slice, we store a hash which is of 32 bytes in length
- For a default slice size of 128 KB (defined by indexes.conf's rawdataChunkSize), we need ~250 KB of extra disk space to store hashes
- Below is the equation:
  - 100 GB / 128 KB = 7812 slices, 7812 * 32 = 249.9 KB
- This holds true for replicated bucket copies in indexer cluster

Securing the Hashes

- By default, splunk writes the hashes to the rawdata directory of every bucket.
- For extra layer of security, customers can backup a copy of hash files when the bucket gets rolled from hot->warm, this can be scriptable.
- Note: Bucket's hash file names are unique for a given index. During backups, you can just store all hash files of an index in one folder.

Migration

- In case of clustering, All the Indexers & Cluster Master should be running 6.3+. Same applies even for standalone instance (6.3+)
- Deals with any new data that is being indexed after turning ON the feature.

Few Points

- Hash computation & storage happens at index time.
- For inflight data coming from forwarders, we trust SSL.
- This is an initial version of the feature and can be extended based on the feedback from the users.

Team

Special thanks to the team members:

- Mustafa Ahamed – Product Manager
- Mitch Blank & Sundar Vasan – Mentors
- Tanusree Sen – QA
- Tameem Anwar – Performance

Q&A

splunk>

THANK YOU

splunk>