Shop Smart At The Kvstore

Best Value Tricks From The Splunk Kvstore And REST API

Duane Waddle

Senior Security Engineer, Defense Point Security

George Starcher

Senior Security Engineer, Defense Point Security

SPLUNK TRUST

.conf2016

splunk>

DEFENSE POINT SECURITY

- George Starcher
  - Splunking since 2010
  - Still in love with the Splunk HEC
  - Favorite game is automating Splunk to earn its keep
- Duane Waddle
  - Splunking since 2010
  - Still dreams of one day being a helicopter pilot
- Our Other .conf Talks:
  - .conf 2015: talks SSL and Advanced Lookups
  - .conf 2014: talks on SSL and Alert Script automation
  - .conf 2016: Duane on: Anti Patterns it seemed like a good idea at the time!

Agenda

- Overview:
  - KVStore basics
  - Why KVStore?
- KV Store Management:
  - Creating a KVStore
  - Backing up KVStore
- Use Cases and Serious Code:
  - Sending Threat Intel into ES Threat Intel KV Store
  - Modular Alert + KVStore = shared lookups across instances
  - Dumb Syncing of Database Tables to KVStore
  - Smart Syncing of Database Tables to KVStore
  - Table Flipping! Driving other table based tech. Routing

All Code has README files and in the github repos linked at the end of the slides.

KVStore - Basics:

Splunk Docs Definition: "The App Key Value Store (or simply, KV Store) feature of Splunk Enterprise provides a way to save and retrieve data within your Splunk apps, thereby enabling you to manage and maintain the state of the application."

- Extension of existing lookup functionality
- MongoDB behind the curtain
- REST api access unlike normal CSV-based lookups
  - Gives us random access changes
- Must run at search head level: tcp 8191 default
- http://dev.splunk.com/view/webframework-features/SP-CAAAEY7
- http://dev.splunk.com/view/SP-CAAAEZJ

splunk> .conf2016

KVStore - Basics:

- In a Search Cluster a KVStore Captain is elected. Might <u>NOT</u> also be the SHC Captain
- All nodes read, Captain handles the writes
- You can get great information from the Distributed Management Console
- Prior to Splunk version 6.3:
  - No auto lookup support
  - No replication to Indexers
- Uses _key hidden field as the unique record key per collection
  - You can specify this like _key=src_ip
  - Otherwise it auto generates a key

splunk> .conf2016

KVStore - Why?

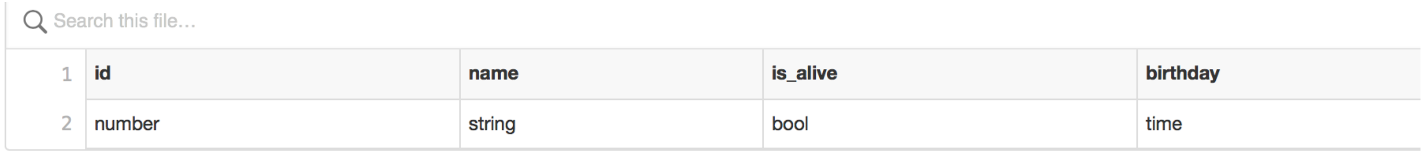
- Splunk is a dynamic indexed data system. Why would it need a random access database?
- Originally added to provide state tracking etc for Splunk App for Enterprise Security

Uses:

- Data value state (ES with notables)
- Lookups: Assets, data enrichment
- Getting sets of structured data out: Tables to control other systems

KVStore - Creation:

- Add by conf file editing/deploying an app. No GUI options for collections.
  - Edit collections.conf and transforms.conf
- Use the REST api:
  - makekvstore.py: https://github.com/georgestarcher/Splunk-createkvstore
    - This makes the collection and it replicates across a cluster
    - This code does not define the lookup in transforms.conf
    - template.csv is the model for your collection
    - The first row defines the field names
    - The second row defines the type of the field



1	id	name	is_alive	birthday
2	number	string	bool	time

KVStore - Creation:

- Edit the kvstore.conf to point to the desired server
- You will provide an admin level Splunk user credential at the prompts when you execute the script

```
> python makekvstore.py app collection
```

- app: is the argument where you specify the app name context you want the collection in. Like “TA-assets”
- collection: is the argument where you specify the collection name to be made. Like “our_assets”

```
> python makekvstore.py TA-assets our_assets
```

splunk> .conf2016

KVStore - Backup: Using Search

```
| inputlookup assets | eval saveKey=_key |
outputlookup kvstore_backup_assets_20160311.csv
```

- Backs up the data not the collection definition.
- Make sure you use a naming convention that is blacklisted from replication!!!

<u>distsearch.conf:</u>

```
[replicationBlacklist]
noBackups = .../kvstore_backup_*
```

KVStore - Backup: Using Python

```
> python backupkvstore.py
```

- Use the REST api:
  - backupkvstore.py: https://github.com/georgestarcher/Splunk-backupkvstore
    - Provide a credential with permissions to all collections to be backed up
    - It will write a text file of JSON data for each collection and the data it contains
    - def_COLLECTIONNAME is the definition
    - data_COLLECTIONNAME is the data

splunk>enterprise

Use Cases and Serious Code

Splunk App for ES Threat Intel

Use Case: I get some list of ips/domains from an email sharing list.

First configure a csv file with the provided template:

```
threat_key,ip,domain,weight,description,address,city,country,
postal_code,state_prov,organization_name,organization_id,
registration_time

manual_intel,10.63.51.1,,low,TEST:IgnoreThreatIncidents,,,,,,,,

manual_intel,
10.63.51.2,,medium,TEST:IgnoreThreatIncidents,,,,,,,,

myisac,
10.63.51.66,scoobydoo.local,high,TEST:IgnoreThreatIncidents,,,,
,,,,
```

Splunk App for ES Threat Intel

> python splunk-es-threat-intel.py -h

```
splunk-es-threat-intel.py -i <inputfile> -c <confile>
--remove
```

- The <inputfile> is the CSV template file from the previous slide.
- The <confile> is the default or a copy of kvstore.conf with the script.
- Edit kvstore.conf with your splunk_server, splunk_user, and a base64 encoded password matching the user for a service account.
- The user must have a Splunk user role that has permissions to write to DA-ESS-ThreatIntelligence where the collections are stored.
- It will log to intel_to_splunk.log where the script is run from.

Splunk App for ES Threat Intel

Preparation: cleanData(data)

- This method preps the data for the endpoint
- It expands out lines with both domain and IP to two entries to submit

Creation/Update: postDataToSplunk(data)

- We call the REST endpoint to create the object
- If it returns an ERROR we assume the object exists. We call the REST endpoint again with the object _key appended to the URL to update the object
- We then call the REST endpoint to create the metadata Threat Group entries

Deletion: removeDataFromSplunk(data)

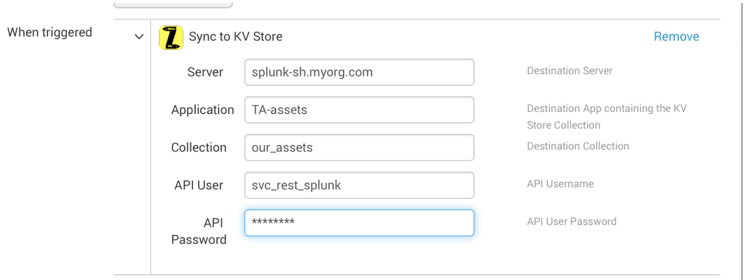
- We just loop through and call the DELETE method on the REST endpoint for that object
- We do NOT remove the Intel Group entries as that could cause old notables to be missing the group metadata info like description

What If I Have Two Different Search Clusters?

- Same search cluster?
  - KVStore auto syncs across search heads in the same cluster
- **Use Case**: I want to sync an asset table over two clusters
  - Try our modular alert: sync_kvstore
  - https://github.com/georgestarcher/sync_kvstore
  - Install the modular alert
  - Configure the alert on a lookup you wish to send TO the destination cluster

```
| inputlookup asset_kvstore
```

- We get around the 10K Modular Alert limit by opening the search results file directly.
- It takes 2.5 minutes to send a 233K row asset table.


When triggered — Sync to KV Store — Remove

Server	splunk-sh.myorg.com	Destination Server
Application	TA-assets	Destination App containing the KV Store Collection
Collection	our_assets	Destination Collection
API User	svc_rest_splunk	API Username
API Password	********	API User Password

Modular Alert - Sync KVStore

First it will clear the remote KV Store collection; then we post the data via threaded batch save call to the API.

```
destKVStore.clearRemoteKVStore()
try:
       postList = []
       for entry in tableContents:
           if ((len(json.dumps(postList)) + len(json.dumps(entry))) < _max_content_bytes) and
(len(postList) + 1 < _max_content_records):
               postList.append(entry)
           else:
               destKVStore.postDataToSplunk(postList)
               postList = []
               postList.append(entry)
       destKVStore.postDataToSplunk(postList)

   except Exception, e:
       raise Exception, "%s" % str(e)
```

DBX Table: Dumb Sync

- **<u>Use Case</u>**: I have a ip360 Vulnerability Scanner system. I want to sync the vuln description table over to splunk
- Splunk DB Connect app lets us do DB lookups (with tears)
  - Search performance impact
  - DB Server performance impact
- Alternative: run a query to fetch a lookup table on a heavy forwarder
  - `| dbquery "<SOME_SQL"> | table ...`
- AND….. use the previously mentioned sync_kvstore modular alert send that DBX sourced lookup table to KVStore lookup in your search heads
- You get to avoid trying to run DBX in a search cluster!!! WIN!!!!!!
- But if your query is big, that could still perform badly
- It does full replace the target table/collection each send

RDBMS Table: Smart Sync

- If caching millions of rows, tears are gonna happen
    - Updating the cache is painful on the DB side (full table scan)
- Can we incrementally update a KVStore collection with data from a DB?
- Need some help in the database (be nice to your DBAs)
    - An auxiliary table to hold “change events”
    - A trigger on the source table that fires on changes and inserts into the auxiliary
- And some glue code
    - Read the auxiliary table to pick up what changes have occurred
    - Push those changed records into KVStore via the REST API

RDBMS Table: Smart Sync

- PostgreSQL Table PRODUCTS:

```
 productid |      description       |     manufacturer      | unitprice | country_of_origin | weight_kg
-----------+------------------------+-----------------------+-----------+-------------------+-----------
         1 | Jar of Dirt            | Capt. Jack Sparrow    |     82.85 | UK                |    1.0000
         2 | Sack of Potatoes       | Pete's Potatoes       |      3.25 | USA               |    2.5000
         3 | Macbook Pro            | Apple Computer        |   1999.95 | China             |    1.1000
         4 | iPad Pro               | Apple Computer        |    999.95 | China             |    0.8000
         5 | 100 Ducks .999 Silver  | Yeager Poured Silver  |      0.65 | USA               |    0.1000
         6 | Fancy Fez              | Splunk                |  10000.00 | USA               |    0.0500
```

- Trigger warning :)

```
CREATE TRIGGER changetrack_products
AFTER INSERT OR UPDATE OR DELETE ON PRODUCTS
    FOR EACH ROW EXECUTE PROCEDURE process_change_products();
```

RDBMS Table: Smart Sync

- Make a DB change:

```
UPDATE PRODUCTS SET manufacturer='Davy Jones'
where manufacturer='Capt. Jack Sparrow';
```

- Trigger picks it up, updates our state table:

```
postgres=# select * from changetrack_products;
 changeid | operation |           stamp            | productid
----------+-----------+----------------------------+-----------
        9 | U         | 2016-07-30 18:31:16.337077 |         1
```

- Script Uses this to push change to KV:

```
2016-07-30 18:31:36 looking for changes newer than 2016-07-30 15:27:38
2016-07-30 18:31:36 Updating kvstore key 1 with record from 2016-07-30 18:31:16.337077
2016-07-30 18:31:36 storing new change state 2016-07-30 18:31:16.337077
```

RDBMS Table: Smart Sync

- Add in some Splunk config options in collections.conf:

```
replicate = true (Splunk v6.3+)
replication_dump_strategy = auto
replication_dump_maximum_file_size = XXX ( def 10240KB)
```

- Result:
  - Usable locally at indexers too
  - Minimized bundle replication impact
- Get the code:
  - https://github.com/georgestarcher/Splunk-smartdbsync-KVStore

⚠️Table Flipping: Routes!

- Wouldn’t it be nice if we could blacklist (from Splunk)
  - Then we wouldn’t have to work so hard
- Most of the time, null routing is as good as a blacklist (possibly better)
  - ACL checks are often done in software, routing in hardware
- What if we could drive a BGP Null Route table via a KVStore collection?
- Make a null-routing server (maybe a Raspi?)

⚠️ Table Flipping: Routes!

- Install Quagga on a Raspberry
- Grab the code project:
- https://github.com/georgestarcher/Splunk-blackhole
  - Edit the settings
  - Crontab the blackholev1.py
  - Edit the table in Splunk KVStore
  - Confirm the routing table edit
  - Redistribute into IGP or BGP

⚠️Table Flipping: Routes!

- Add some data to the KV store collection via SPL

```
| makeresults | eval cidr="13.14.15.16/32", blackhole="true" | eval time=_time | appendcols [ rest /services/authentication/current-context/context/ | fields username, email ] | inputlookup append=true blackhole | stats first(*) as * | outputlookup blackhole
```

- Run the script to add/remove routes in Quagga

```
$ ./blackholev1.py
2016-07-31 13:48:55 Add blackhole for 13.14.15.16/32 successful.  Requestor=admin at=1470023301
```

Code Link Collection:

- Create KVStore: https://github.com/georgestarcher/Splunk-createkvstore
- Backup KVStore: https://github.com/georgestarcher/Splunk-backupkvstore
- Sync KVStore: https://github.com/georgestarcher/sync_kvstore
- ES Intel: https://github.com/georgestarcher/Splunk-ESIntel-KVStore
- Smart DB Sync: https://github.com/georgestarcher/Splunk-smartdbsync-KVStore

What Now?

Other breakouts with Splunk Trust members:

- Fields, Indexed Tokens and You - Martin Müller
- Architecting Splunk for Epic Performance at Blizzard Entertainment - Mason Morales
- Lesser Known Search Commands - Kyle Smith
- Best Practices for Aggregating and Grouping Data From Different Search Results - Nick Mealy

THANK YOU

.conf2016

splunk>