CLI commands for starting, stopping, status, etc.	
Manage Splunk processes	`splunk [start | stop | restart]`
Start and automatically accept the license without prompt	`splunk start --accept-license`
Enable boot start on Linux where xyz is the name of the user account. This command *must* be run as root	`splunk enable boot-start -user xyz`

Display a usage summary for help, plus various other help options	`splunk help`
	`splunk help cluster`
	`splunk help shcluster`
	`splunk help add`
	`splunk help show`

Splunk version	`splunk version`
Splunk running status	`splunk status`
Splunk Web port	`splunk show web-port` `splunk set web-port port#`
Splunk management (splunkd) port	`splunk show splunkd-port` `splunk set splunkd-port port#`
Splunk App Server ports	`splunk show appserver-ports` `splunk set appserver-ports port#`
Splunk KV store port	`splunk show kvstore-port` `splunk set kvstore-port port#`
Splunk server name	`splunk show servername` `splunk set servername name`
Default host name	`splunk show default-hostname` `splunk set default-hostname name`
Show the guid of this instance	`splunk show guid`

CLI commands for licensing	
On the master license server, add a new license	`splunk add licenses \` `absolutepathtolicensefile`
On the master license server, list the licenses	`splunk list licenses`
Make this instance a license slave of a master	`splunk edit licenser-localslave \` `-master_uri https://Lic_Master:port`
List license status of this instance	`splunk list licenser-localslave`
List all license slaves (run on license master)	`splunk list licenser-slaves`
List any license alerts or warnings	`splunk list licenser-messages`
List current license groups	`splunk list licenser-groups`
Change the active license group (e.g., to change to Forwarder group)	`splunk edit licenser-groups group \` `-is_active 1`

CLI commands for general administration	

Create a user	`splunk add user name \`
	`-password "password" \`
	`-full-name 'User Name' -role role_name`

Change a user's password	`splunk edit user name \` `-password newpassword`
Remove a user	`splunk remove user name`
Create a role	`splunk add role role_name \` `-imported other_role_name`
On a search head, add a distributed search peer	`splunk add search-server peer:port \` `-remoteUsername user -remotePassword pass`
Display information about the search job scheduler (run on search head)	`splunk show scheduler-status`

Move search jobs from dispatch directory based on the last modification time of the job; `mod_time` is a relative time in SPL format	`splunk cmd splunkd clean-dispatch dest_directory mod_time`
	Example:
	`splunk cmd splunkd clean-dispatch /tmp/jobs/ -7d@d`

CLI commands for inputs	

Set up an input
There are many options; some are required
[Note: exec is scripted input
oneshot is a batch input]
	`splunk add monitor file_or_dir`
	`splunk add tcp port`
	`splunk add udp port`
	`splunk add exec script_to_run`
	`splunk add oneshot file_or_dir`

Show the automatic sourcetype that Splunk will assign to this input	`splunk test sourcetype file_to_test`

Identify what Splunk is monitoring:
files and directories
local and remote event logs, perfmon
status of inputs
	`splunk list monitor`
	`splunk list wmi`
	`splunk list eventlog`
	`splunk list perfmon`
	`splunk list inputstatus`
	`splunk list exec`

CLI commands for indexes	
Create an index	`splunk add index indexName`
Remove all data from an index (run on indexer)	`splunk clean eventdata \` `[ -index indexName ]`
Remove all data from the kvstore	`splunk clean kvstore \` `-collection collection_name`

Remove the file pointer for a particular source from the fishbucket, so the file will be re-indexed	`splunk cmd btprobe -d \`
	`SPLUNK_HOME/var/lib/splunk/`
	`fishbucket/splunk_private_db \`
	`--file source --reset`

Recreate the idx files for a bucket	`splunk rebuild path_to_bucket`
Reload the index configurations	`splunk reload index`
When using data integrity: check an index	`splunk check-integrity \` `-index indexName verbose`
When using data integrity: check a bucket	`splunk check-integrity \` `-bucketPath path_to_bucket verbose`

When using data integrity: regenerate hash files (either for a bucket or for an entire index)	`splunk generate-hash-files \`
	`-bucketPath path_to_bucket`
	`splunk generate-hash-files \`
	`-index indexName`

CLI commands for apps	
Install an app from the named file on the server	`splunk install app appfile`
Package an app	`splunk package app appname`
Shows the status of an app, whether it is installed or not, enabled/disabled, or visible/invisible	`splunk display app appfolder`
Remove an installed app from this server	`splunk remove app appfolder`
Create a new (empty) app, where the template can be `barebones` or `sample_app`	`splunk create app appname \` `–template template_name`
CLI commands for debugging	
Display the merged on-disk configurations for a configuration type (e.g. `inputs`)	`splunk show config conf_name`
Check or display the configs for a type (see more information on btool at the end of this document)	`splunk btool check` `splunk btool conf_name list [ --debug ]`
Display the status of an app	`splunk display app appdirname`
Test your regular expression (see example at end of this document)	`splunk cmd pcregextest`
CLI commands for forwarding/receiving and deployment server	
Sets a receiving port `rport` (run on indexer)	`splunk enable listen rport`
On an indexer, shows all configured receiving ports	`splunk display listen`
Forward inputs to the indexer (`idx`) that is listening on port `rport` (run on forwarder)	`splunk add forward-server idx:rport`
On a forwarder, show where it is sending its inputs	`splunk list forward-server`

On a forwarder, remove a configured target indexer	`splunk remove forward-server idx:rport`
On any non-clustered instance, set the instance to use the deployment server (`dserver`)	`splunk set deploy-poll dserver:port`

On any instance, check its deployment client/server status; deploy-poll shows the server:port that the client is contacting	`splunk show deploy-poll`
	`splunk display deploy-server`
	`splunk display deploy-client`

On the deployment server, list all clients	`splunk list deploy-clients`
On the deployment server, reexamine all deployment apps	`splunk reload deploy-server`
CLI commands for indexer clustering	
Single Site	

Make this instance a cluster master	`splunk edit cluster-config \`
	`-mode master -replication_factor 2 \`
	`-search_factor 2 -secret mycluster`

Make this indexer a cluster peer	`splunk edit cluster-config -mode slave \`
	`-master_uri https://master:port \`
	`-secret mycluster -replication_port 9000`

Give this search head the ability to search a cluster	`splunk edit cluster-config \`
	`-mode searchhead \`
	`-master_uri https://master:port \`
	`-secret mycluster`

Give this search head the ability to search an *additional* cluster	`splunk add cluster-master \`
	`-master_uri https://master:port \`
	`-secret cluster2`

Restart all peers from the master	`splunk rolling-restart cluster-peers`
Multisite	

Make this instance a cluster master of a multisite cluster	`splunk edit cluster-config \`
	`-mode master -multisite true \`
	`-site site1 \`
	`-available_sites site1,site2 \`
	`-site_replication_factor origin:1,total:2 \`
	`-site_search_factor origin:1,total:2 \`
	`-secret mycluster`

Make this indexer a cluster peer in a multisite cluster	`splunk edit cluster-config \`
	`-master_uri https://master:port \`
	`-mode slave -site site1 \`
	`-replication_port port -secret mycluster`

Give this search head the ability to search a multi-site cluster	`splunk edit cluster-config \`
	`-mode searchhead -multisite true \`
	`-master_uri https://master:port \`
	`-site site1 -secret mycluster`

Restart all peers from the master
(site by site is optional)
	`splunk rolling-restart cluster-peers \`
	`[ -site-by-site true`
	`-site-order site2,site1,site3 ]`

General Indexer Cluster Commands	
Put cluster in maintenance mode (run on master)	`splunk [ enable | disable | show ] \ maintenance-mode`
Stop this peer gracefully. With enforced counts, takes peer offline permanently, otherwise peer must restart within 60 seconds.	`splunk offline [--enforce-counts]`
Change the length of time before an offlined peer must restart	`splunk edit cluster-config \ -restart_timeout seconds`
Replicate report acceleration and data model acceleration summaries (run on master)	`splunk edit cluster-config \ -summary_replication true`
Assign a label to all the search heads and peers that are part of this cluster (run on master)	`splunk edit cluster-config \ -cluster_label label_name`
Apply cluster-master apps to all peers (run on master)	`splunk apply cluster-bundle`
Show status of bundle deployment (run on master)	`splunk show cluster-bundle-status \ [--verbose]`
Show cluster status (run on master)	`splunk show cluster-status [--verbose]`
Remove offline peers entirely from the cluster (run on master)	`splunk remove cluster-peers \ -peers guid1,guid2`
List excess buckets	`splunk list excess-buckets [index]`
Remove excess buckets	`splunk remove excess-buckets [index]`
Allow searching to begin before RF is met (run on master)	`splunk set indexing-ready`
Run diag from the cluster master	`splunk diag --enable=rest`
Rebalance primaries (see also REST ENDPOINTS at end of document)	`https://yourCM:mgmtport/services/cluster/master/control/control/rebalance_primaries`

Perform data rebalancing on the cluster or a specific index, optionally setting a maximum run time	`splunk rebalance cluster-data \ -action start [-index index] \ [-max_runtime minutes]`
	
	`splunk rebalance cluster-data \ -action status`
	
	`splunk rebalance cluster-data \ -action stop`

Set the threshold for data rebalancing, where 1.0 would be "fully balanced"	`splunk edit cluster-config \ -rebalance_threshold 0.90`

Get various information about the indexer cluster	splunk list cluster-config splunk list cluster-master splunk list cluster-peers splunk list master-info splunk list cluster-buckets splunk list peer-info splunk list peer-buckets

CLI commands for search head clustering	
Initialize a search head when creating a SH cluster	splunk init shcluster-config \ -mgmt_uri https://thisSH:port \ -replication_port port -secret cluster2
Manually assign a captain and set a member list (run on the new captain)	splunk bootstrap shcluster-captain \ -servers_list https://SH2:port, \ https://SH3:port,https://SH4:port
Clean the dynamic configuration files for a member (run on the member with problems)	splunk clean raft
Add this search head to an existing SH cluster (run on the new member)	splunk add shcluster-member \ -current_member_uri \ https://existingmember:port
Add a new search head to an existing SH cluster (run from any current member)	splunk add shcluster-member \ -new_member_uri https://new_member:port
Configure a SHC member to access the deployer	splunk edit shcluster-config \ -conf_deploy_fetch_url \ https://deploy_server:port
Help a SHC member get back in sync	splunk resync shcluster-replicated-config
Show the status of the SH cluster (run on any member)	splunk show shcluster-status
Show the members of the SH cluster (run on any member)	splunk list shcluster-members
Restart all members of the SH cluster	splunk rolling-restart shcluster-members
Show the status of a rolling restart	splunk rolling-restart shcluster-members \ -status 1
Designate a captain and turn off dynamic election (run on captain)	splunk edit shcluster-config \ -election false -mode captain \ -captain_uri https://captain:port
Designate a captain and turn off dynamic election (run on members)	splunk edit shcluster-config \ -election false -mode member \ -captain_uri https://captain:port
Convert SHC members to dynamic election mode (run on all members, run on static captain last, then bootstrap)	splunk edit shcluster-config \ -election true \ -mgmt_uri https://this_member:port
Install app bundles on all SH cluster members (run from deployer)	splunk apply shcluster-bundle \ -target https://existingmember:port

Set a label for the SH cluster in the DMC for reporting; Run this on any member and on the deployer	`splunk edit shcluster-config \` `-shcluster_label label_name`
Permanently disable SH clustering on this instance	`splunk disable shcluster-config`
Remove this SH cluster member from the cluster (run on the member)	`splunk remove shcluster-member`
From another instance, remove a SH cluster member (The mgmt_uri is the member to be removed)	`splunk remove shcluster-member \` `-mgmt_uri https://thatSH:port`

Get various information about the SH cluster	`splunk list shcluster-config`
	`splunk list shcluster-members`
	`splunk list shcluster-captain-info`
	`splunk list shcluster-artifacts`
	`splunk list shcluster-scheduler-jobs`
	`splunk list shcluster-member-info`
	`splunk list shcluster-configuration-set`
	`splunk list shcluster-member-artifacts`

Run diag from the SH cluster captain	`splunk diag`

Notes:

In most Linux environments (depending on the PATH), the splunk command must be prefixed with "./"

```
./splunk start
```

All commands are written on a single line, even when they are shown on multiple lines. Cut and paste may not work properly from this document because of this. To make cut-and-paste work better, the Linux line-continuation character "\" has been added at the end of each line; do not include this character when manually typing the command on a single line!

REST ENDPOINTS

You can use REST endpoints instead of many CLI commands. However, the purpose of this section is to capture the some of the REST endpoints for which no CLI equivalent exists. Documentation for each endpoint can be found in the Splunk REST API Reference Manual. [http://docs.splunk.com/Documentation/Splunk/latest/RESTREF]

Endpoints can be accessed via the REST API directly using toolls such as curl, or by putting the endpoint into a browser, like this
`https://<host>:<mPort>/services/endpoint`
where host is the Splunk host and mPort is the splunkd port (aka management port). Note that you must use https to access the splunkd port. You will typically need to authenticate with an admin account and password to proceed.

In addition to accessing the REST API directly, you may choose to download an SDK and use a higher-level library in your code. See http://dev.splunk.com for more details about the REST API and the SDKs, including tutorials and user guides.

Function	Endpoint
Indexer Cluster	
Initiate primary rebalancing manually for an indexer cluster	cluster/master/control/control/rebalance_primaries

Function	Endpoint
View the number of primaries on a peer	cluster/master/peer
Adjust cluster peer detention mode	cluster/slave/control/control/set_detention_override
Re-add the cluster peer (indexer) to the cluster master	cluster/slave/control/control/re-add-peer
Search Head Cluster	
Access configuration replication health statistics for SHC	replication/configuration/health
Lists searchhead cluster artifacts and replicas (must run on captain)	shcluster/captain/artifacts
General	
See the current in-memory configuration (like btool)	properties

Cluster endpoint descriptions:
http://docs.splunk.com/Documentation/Splunk/6.5.1/RESTREF/RESTcluster

btool Supplement

btoool displays merged **on-disk** configuration values. It is a helpful tool for finding basic configuration problems. (Some of the btool commands are also listed in the tables above.)

- To quickly check the syntax of all configuration files on an instance:

  ```
  splunk btool check
  ```

- To list the configurations of a single type, use the following form of btool. Substitute the name of the configuration file (without the .conf extension) for `conf_name` in the command:

  ```
  splunk btool conf_name list [ --debug ]
  ```

- To see a single stanza, you can include the stanza, for example:

  ```
  splunk btool inputs list monitor:///var/log
  ```

  However, the command line must specify the stanza exactly in order to match.
  You can also specify the user and app to see the configurations from a user point of view.
  If you specify the user, you must also specify the app

  ```
  splunk btool conf_name list [ --user=user_name --app=app_name ]
  ```

As an alternative to btool, you can see the current **in-memory** configuration values with

`https://host:mPort/services/properties/`

where `host` is the name of the indexer and `mport` is the management port.

pcregextest

This is a command line tool to test a regular expression. You must give the tool the regular expression to test, and a test string to test against. For example:

```
./splunk cmd pcregextest \
mregex='(?<src_ip>\d+(?:\.\d+){3})' test_str="1.1.1.1 2.2.2.2"
```

Both `mregex` and `test_str` are required.

Using the CLI to manage the HTTP Event Collector

http://dev.splunk.com/view/event-collector/SP-CAAAE7D