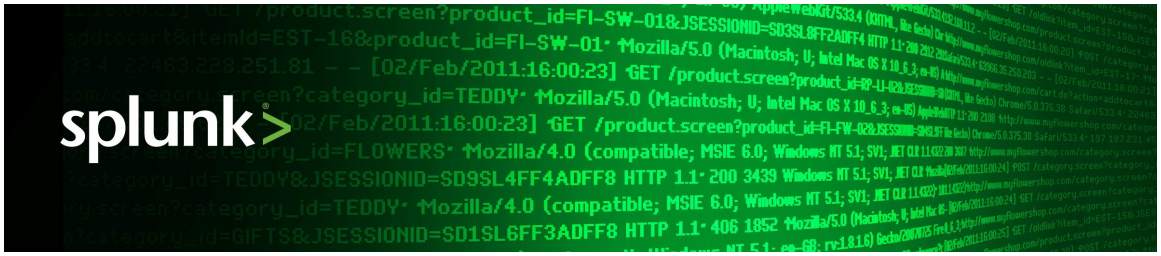


Splunk® Universal Forwarder Forwarder Manual 6.5.0

Configure load balancing for Splunk Enterprise

Generated: 10/19/2016 9:56 pm

Configure load balancing for Splunk Enterprise

With **load balancing**, a forwarder distributes data across several receiving instances. Each receiver gets a portion of the total data, and together the receivers hold all the data. To access the full set of forwarded data, you need to set up distributed searching across all the receivers. See "About distributed search" in *Distributed Search*.

Load balancing enables horizontal scaling for improved performance. In addition, its automatic switchover capability ensures resiliency in the face of machine outages. If a host goes down, the forwarder sends data to the next available receiver.

Load balancing can also be of use when you get data from network devices like routers. To handle syslog and other data generated across TCP port 514, a single universal forwarder can monitor port 514 and distribute the incoming data across several indexers.

Note: You should not use an external load balancer to implement load balancing between forwarders and receivers. This practice does not generate the results you would expect. Use the load balancing capability that comes with the forwarder.

How load balancing works

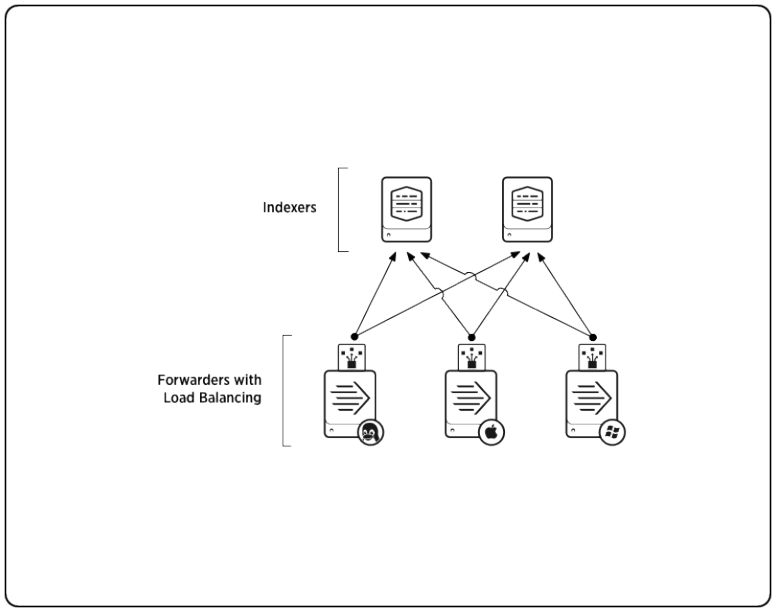
Forwarders perform automatic load balancing. The forwarder routes data to different indexers on a specified time interval. For example, if you have a load-balanced group that consists of indexer A, B, and C, at a specified interval, the forwarder switches the data stream to another indexer in the group at random. The forwarder might switch from indexer B to indexer A to indexer C, and so on. If one indexer is down, the forwarder immediately switches to another.

There is a data stream for each of the inputs that the forwarder monitors. The forwarder determines if it is safe for a data stream to switch to another indexer. Then, at the specified interval, it switches the data stream to the newly selected indexer. If it cannot switch the data stream to the new indexer safely, it keeps the connection to the previous indexer open and continues to send the data stream until it has been safely sent.

Universal forwarders cannot switch indexers when they monitor TCP network streams of data unless the forwarder encounters an end-of-file (EOF) or the receiving indexer goes down. At that point, the forwarder switches to the next indexer in the list. Because the universal forwarder does not parse data and identify event boundaries before forwarding the data to the indexer (unlike a

heavy forwarder), it does not know when it is safe to switch to the next indexer unless it receives an EOF.

The following diagram shows a typical load-balancing scenario, in which three forwarders are sending load-balanced data across a set of two receiving indexers:



Options for configuring receiving targets for load balancing

When you configure a set of target receivers for load balancing, you can employ either DNS or static lists.

DNS lists provide greater flexibility and allow for simplified scaling, particularly for large deployments. Through DNS, you can change the set of receivers without needing to re-edit each forwarder `outputs.conf` file.

Static lists let you specify a different port for each receiver. This is useful if you need to perform load balancing across multiple receivers that run on a single host, as each receiver can listen on a separate network port.

Specify a static list target

1. On a forwarder that you want to set a static list target, edit `$SPLUNK_HOME/etc/system/local/outputs.conf`. You might have to create this file beforehand.
2. In the `outputs.conf` file, specify each of the receivers in the target group `[tcpout]` stanza.

3. Save the `outputs.conf` file.
4. Restart the forwarder. The forwarder sends data to the static list targets.

In the following example, the target group consists of three receivers, specified by IP address and receiver port number. The universal forwarder balances load between the three receivers. If one receiver goes down, the forwarder switches to another one on the list.

```
[tcpout: my_LB_indexers]
server=10.10.10.1:9997,10.10.10.2:9996,10.10.10.3:9995
```

Specify a DNS list target

1. On a forwarder that you want to set a DNS list target, edit `$SPLUNK_HOME/etc/system/local/outputs.conf`. You might have to create this file beforehand.
2. In the `outputs.conf` file, specify a single host in the target group `[tcpout]` stanza.
3. Save the `outputs.conf` file.
4. Restart the forwarder. The forwarder sends data to the DNS list targets.
5. On your DNS server, create a DNS A record for each host IP address, referencing the server name you specified in `outputs.conf`..
   ```
   splunkreceiver.mycompany.com   A   10.10.10.1
   splunkreceiver.mycompany.com   A   10.10.10.2
   splunkreceiver.mycompany.com   A   10.10.10.3
   ```
6. Reload the updated configuration on your DNS server. It might take a while for DNS changes to take effect, depending on the size of your network topology.

In the following example, the forwarder has been configured to send data to what appears to be a single host. The changes you made in DNS now have this hostname refer to three different IP addresses.

```
[tcpout:my_LB_indexers]
server=splunkreceiver.mycompany.com:9997
```

The forwarder uses the DNS list to load balance, sending data in intervals, switching among the receivers specified. If a receiver is not available, the forwarder skips it and sends data to another one on the list.

If you have a topology with many forwarders, the DNS list method lets you update the set of receivers by making changes on the DNS server, without having to edit `outputs.conf`.

Configure universal forwarder load balancing for horizontal scaling

When you configure load balancing for horizontal scaling, you should first determine your needs, particularly your horizontal scaling and whether or not you need failover. This helps you develop a topology based on those needs, which can include multiple forwarders as well as receivers and a search head to search across the receivers.

Set up DNS-based load balancing

This procedure assumes a topology of three universal forwarders and three receivers and uses a DNS list to designate the receivers. The receivers must all listen the same port.

1. Install a set of three Splunk Enterprise instances as receivers.
2. Configure receiving on the receivers. Specify the same receiving port. For example:
   ```
   ./splunk enable listen 9997 -auth <username>:<password>
   ```
3. Install a set of universal forwarders, as described in Install the universal forwarder software.
4. On your DNS server, set up a DNS list with an A record for each receiver IP address.
   ```
   splunkreceiver.mycompany.com   A   10.10.10.1
   splunkreceiver.mycompany.com   A   10.10.10.2
   splunkreceiver.mycompany.com   A   10.10.10.3
   ```
5. Reload the updated configuration on your DNS server. It might take a while for DNS changes to take effect, depending on the size of your network topology.
6. Create an `outputs.conf` file for all the forwarders to use. This example specifies the DNS server name used in the DNS list and the port the receivers are listening on.
   ```
   [tcpout]
   defaultGroup=my_LB_indexers

   [tcpout:my_LB_indexers]
   disabled=false
   autoLBFrequency=40
   server=splunkreceiver.mycompany.com:9997
   ```

   This `outputs.conf` file uses the `autoLBFrequency` attribute to set a load-balance frequency of 40 seconds. Every 40 seconds, the forwarders switch to another receiver. The default frequency is 30 seconds.
7. Distribute the `outputs.conf` file to all the forwarders. You can use the **deployment server** to handle the distribution.

Specify load balancing from the CLI

You can also use the CLI to specify load balancing. You do this when you start forwarding activity to a set of receivers.

```
./splunk add forward-server <host>:<port> -method autobalance
```

where `<host>:<port>` is the host and receiver port of the receiver.

This example creates a load-balanced group of four receivers:

```
./splunk add forward-server indexer1:9997 -method autobalance
./splunk add forward-server indexer2:9997 -method autobalance
./splunk add forward-server indexer3:9997 -method autobalance
./splunk add forward-server indexer4:9997 -method autobalance
```

Configure improved load balancing with props.conf

You can improve how the universal forwarder distributes data during load balancing by using the `props.conf` configuration file. In the file, you specify settings to enable use of a special processor called `ChunkedLBProcessor` that distributes data more evenly amongst the indexers in a load-balanced target group. This configuration can be enabled for any source type. The processor and these settings are available on universal forwarders that are version 6.5.0 and later only.

This feature works with any kind of load balancing setup. It controls how the forwarders package and send the data to the receivers.

1. Set up your forwarders and receivers for load balancing, as described earlier in this topic.
2. On the forwarders where you want to improve data distribution, edit `$SPLUNK_HOME/etc/system/local/props.conf`. You might have to create this file beforehand.
3. In the `props.conf` file, add a stanza for the source type where you want to improve data distribution. You do not need to perform this step if the source type stanza already exists.
4. In the stanza, set the `EVENT_BREAKER_ENABLE` setting to `true`.
5. (Optional) Add and configure the `EVENT_BREAKER` setting to a regular expression that represents the event boundary. The forwarder uses this expression to determine when it is okay to change between receivers in a load balancing configuration.
6. Save the `props.conf` file and close it.

7. Restart the universal forwarder.

Props.conf settings to improve distribution of data in load balancing

The settings for using the `ChunkedLBProcessor` processor on the universal forwarder are as follows.

Attribute	Description	Default
`EVENT_BREAKER_ENABLE`	Enables the use of the `ChunkedLBProcessor` data processor, which improves distribution of data from universal forwarders to receiving indexers for a given source type. A value of true tells the forwarder to use the `ChunkedLBProcessor` processor for the source type. A value of false tells the forwarder to send data to the receiver through standard load balancing methods.	`false`
`EVENT_BREAKER`	A regular expression that specifies the event boundary for the universal forwarder to use to determine when it can send events to the indexer. The regular expression must contain a capturing group (a pair of parentheses that defines an identified sub-component of the match.) This setting is similar to the `LINE_BREAKER` setting for heavy forwarders and indexers, except that the forwarder does not discard the contents of the capturing group when it encounters a match. The forwarder looks for the `EVENT_BREAKER` regular expression pattern match in the data stream and uses the match to identify the event boundaries. When it finds a match, the forwarder considers the first capturing group to be the end of the previous event and the end of the capturing group to be the beginning of the next event. The forwarder can then change the receiving indexer based on	`\r\n` (the standard event breaking string)

these event boundaries.

This setting works best with multiline events.

Troubleshoot line merging problems for forwarded custom source types at the indexer

When you configure these settings on the forwarder in conjunction with a custom source type, it is possible that events could be merged incorrectly on the indexer. If this happens, you can configure `props.conf` on the indexer with the `SHOULD_LINEMERGE` setting for the affected source type. If the source type mainly generates single line events, set `SHOULD_LINEMERGE` to `false` for the source type on the indexer.

For example, if you have a custom source type `json_logs` and you notice that the events for the source type appear to be merged incorrectly when you search the source type on your indexers, edit the source type on the indexer to prevent this behavior.

1. On the indexer, open `$SPLUNK_HOME/etc/system/local/props.conf` for editing.
2. In the file, locate the `json_logs` stanza. If this stanza does not exist, create it.
3. Add the `SHOULD_LINEMERGE=true` entry to the stanza.

   ```
   [json_logs]
   SHOULD_LINEMERGE=false
   ```
4. Save the `props.conf` file and close it.
5. (Optional) Add the stanza to `props.conf` on all other indexers.
6. Restart Splunk Enterprise on the indexers.

Example of ChunkedLBProcessor usage

The following example turns on the processor and uses the standard event boundary to determine when to send events.

```
[mysourcetype]
EVENT_BREAKER_ENABLE=true
```

The following example uses the processor to break up events for a source type that generates multiline Java event logs.

```
[mysourcetype2]
```

```
EVENT_BREAKER_ENABLE=true
EVENT_BREAKER=([\r\n]+)(\d\d\d\d-\d\d-\d\d \d\d?:\d\d:\d\d)
```