CloudVyzor ElasticSearch Backup Exporter v1.2

CloudVyzor ElasticSearch Backup Exporter tool (ESBackupExporter.exe) is designed to export logs from your ElasticSearch backup storage into compressed text files (*note: only *.gz format is currently supported*), without restoring it into ES cluster. The tool supports Azure container, Amazon S3 bucket and file share or local folder both for ES backup and for the export locations, and allows both full and incremental export. As compressed logs usually take 5-20 times less space than Elasticsearch backup, the tool can be scheduled on a regular basis to store compressed backups in human-readable format, allowing to keep more historic logs. Also, per-index and per-log message filtering allows to use this tool to create historical reports, exporting only logs of interest.

After the backup is exported, you can use any tool that reads .gz files, or use CloudVyzor Logpad self-hosted edition to get the best experience working with logs.

Configuring ES backup and export locations.

Before using the tool, backup and export folders should be configured. Open ESBackupExporter.exe.config and specify repository and output properties:

```
<add key="repository" value="elasticsearch-backup-location" />
<add key="output" value="exported-files-location" />
```

These properties are in Cloudvyzor Logpad format and support the following locations:

Amazon S3 bucket:

type=s3;key=*s3-storage-key***;secret=***s3-storage-secret***;region=***s3-region***;bucket=***s3-bucket*

Example:

type=s3;key=ABCDEF;secret=1234567890;region=us-west-1;bucket=production-logs-backup

Azure Container:

https://azure-storage**.blob.core.windows.net/**azure-container**?***shared-access-parameters*

Example:

https://prodstorage.blob.core.windows.net/logs-backup?sv=2022-01-01&si=readonly&sr=c&sig=12345

Note: only shared access signature (SAS) URLs are supported for Azure containers for security reasons.

Local folder or share:

Examples:

e:/es-backup

\\backupsrv\logs\prod\

Warning: When this tool is run against backup and/or export locations in Azure or Amazon S3, it is very important to run it on a virtual machine inside the same azure data center or amazon region, to avoid data transfer costs! If you use CloudVyzor Logpad to work with the exported logs, also make sure you run it in the same data center/region.

Running the tool

Before the actual export, you can run the tool in test mode to review and filter out indexes you don't need to export:

ESBackupExporter.exe -test

Note 1: usually, indexes starting with dot '.' are elasticsearch/logstash/kibana system indexes which you might consider to ignore during export. Next section explains how to configure index filtering and other export options.
Note 2: you can export all indexes leaving default filter (''), but it might slow down the export*

To run the actual export, just run the tool without any parameters:

ESBackupExporter.exe

The tool will read all JSON documents from Elasticsearch backup and write them to the compressed text files, json document per line, one or more files per ES index. The tool will print the progress, along with basic per-index statistics, to the console output, and write more detailed progress logs into file exportlog.txt.
Note: if the tool is run for the first time, it might take long time to complete (depending on your server capabilities)

Configuring export options

The tool provides reasonable defaults that you can start working with, but allows to configure several options depending on your server configuration. Here is the list of options you can control:

```
<add key="threads" value="8" />
```

Specifies number of threads to do export with. As performance grows linearly with number of threads used, it is usually recommended to have as many threads as number of processors on your machine, but you can decrease it if server has not enough RAM available or low network throughput.

```
<add key="memoryCachePerThreadMB" value="256" />
```

Memory available per-thread to process and sort ES events before saving them to the file. Increasing it will result in faster export but would require more memory. With the default setting, memory usage may vary, but the tool usually requires 0.5 – 1 GB of RAM **per thread**.

```
<add key="repository" value="es-backup-storage" />
```

Source storage where the ES backup lives

```
<add key="output" value="export-location" />
```

Destination storage where compressed .gz files with exported logs will be stored

```
<add key="query" value="*" />
```

Filters individual events. The filter is in Cloudvyzor Logpad format (xml-encoded):

Simple word:	error
Word with wildcards:	10.*.*.27
Phrase:	"server error"
Phrase with wildcards:	"Error: *exception"
Exclude filter:	-Debug
AND clause (space-separated):	error exception

Note: double-quotes must be encoded as ' in the xml format.

Example:

```
<add key="query" value="error -&amp;apos;client error&amp;apos;" -debug />
```

Note 1: you can use filter option to make historical reports over ES backup instead of full export.
Note 2: Leave default value of '' to export all logs*.
Note 3: OR clause is coming in the next release

```
<add key="indexNameFilter" value="*" />
```

Filters ES indexes by name. The filter is also in Cloudvyzor Logpad format.

Example:

```
<add key="indexNameFilter" value="-.kibana* -logstash* -*2018*" />
```

This query filters indexes that don't start with ".kibana" or "logstash" and don't contain "2018" year in it.

Note 1: you can use this option to exlude system indexes which usually start with '.' specifying

```
<add key="indexNameFilter" value="-.*" />
```

Note 2: Leave default value of '' to export all ES indexes*.

```
<add key="verbose" value="false" />
```

In case of errors or other problems (for example, memory or performance related) set this option to true, so that the tool will write more detailed logs into file exportlog.txt.
Note: enabling this option will cause exportlog.txt to grow very large

```
<add key="incrementalExport" value="true" />
```

If enabled, the tool skips already exported indexes when run second time, provided that the indexes were not changed. It is useful when scheduling regular export, or to re-export only subset of indexes, e.g. when previous export failed due to network problems. Turn it off only in special cases, e.g. when receiving newer version of the tool which fixed incorrect export in the previous version.

```
<add key="sortEvents" value="true" />
```

As log messages are not sorted in Elasticsearch indexes, the tool sorts them as they are collected, and writes them in parts of a fixed size (depending on the memoryCachePerThreadMB setting). After all messages are processed, if this option is set to true, the parts are merged together into one or more files where messages are sorted by timestamp throughout, and duplicates are removed. If this option is turned off, the parts are left in the export without merging them together. Note that messages inside these parts are still sorted, but the parts may have intersecting timestamp ranges. Also, deduplication will not work, so two parts can contain the same event (depending on the other settings). Also, turning this option off makes the tool to run 2-5 times faster and use up to 2 times less memory.
Note: if you are ok with exported index files having intersecting timestamp ranges, and your indexes were never modified (i.e. only new messages appended), then the following combination of parameters will give you much better performance: sortEvents=false; deduplicate=false; parshShardIndex=true; lastSnapshotOnly=true.

```
<add key="deduplicate" value="true" />
```

As Elasticsearch backup can contain several snapshots, there can be multiple copies of log messages in the backup. This option allows the tool to leave only one copy of such messages, if several snapshots are exported. Usually this option should not be turned off, as it only improves performance by a few percent.
Note: for de-duplication to work correctly, sortEents option must be turned on.

```
<add key="parseShardIndex" value="true" />
```

This option allows the tool to parse Elasticsearch backup configuration files, to avoid reading unnecessary parts of the backup files. If turned off, the tool will read all ES backup data, increasing network usage by a factor of two. If the configuration files cannot be parsed (e.g. due to unsupported format), the tool will fall back to reading all data files automatically, so turn off this option only if you suspect that the tool reads these configuration files incorrectly.

```
<add key="lastSnapshotOnly" value="false" />
```

ES backup can contain several snapshots per index. Logs in ES are deleted per-index, not per-message, so last snapshot usually contains all data that the previous snapshots have, so turn this option on to only export last snapshot and skip all previous snapshots. However, if the index was deleted between snapshots or some log messages are manually deleted (using ES API), then previous snapshots can contain data that the last snapshot does not have. This might also happen if last snapshots are in partial or failed state. This option is turned off by default, because, although turning it on usually increases performance by 5-40%, it can result in missing log messages in the export. Turn it on only if you are sure that the messages were not deleted between snapshots or if you are only interested in the latest snapshot and your snapshots always succeed.

Limitations

The tool is designed to provide the best performance possible, so it does not use Elasticsearch API; it instead parses the backup files and extracts json documents using its own code. The tool was tested against backups of Elasticsearch 7.4, but if newer versions of Elasticsearch change the data format, the tool might start working incorrectly. Please contact CloudVyzor support if you want to support other versions of Elasticsearch.

The tool does its best to extract timestamp from logs, but it is sometimes tricky to determine which timestamp is the correct one, because logs usually have several different timestamps. Currently, it supports basic variations of ISO 8601 date and time format, e.g. 2020-01-01T00:00:00.000. Please contact CloudVyzor if you want it to support more formats.
The tool tries to locate the timestamp in the following order:

- Searches for field “@timestamp” which is usually used by Elasticsearch/Logstash
- Searches for first field which contains timestamp in the expected format
- Searches for timestamp in the expected format anywhere in the log message.

Please contact CloudVyzor support if your logs use different timestamp pattern.

Performance

Note: numbers in this section might differ considerably from your case

With the default settings and with ES backup and export folder in Azure or Amazon, the tool usually processes 100-200GB of ES backup files per hour, and produces 5-20GB of exported index logs. The best way to increase performance is to use server with more CPU (usually, network throughput is not a bottleneck in Azure/Amazon).

Example 1:

ES backup contains 100 indexes and 200 GB of total data in an amazon bucket

Tool is run on a server with 4 CPU cores and 8 GB of memory and the default settings

Full export takes 2 hours and produces 25GB of compressed text files.

Example 2:

ES backup contains 10000 indexes and 25 TB of total data in an amazon bucket

Tool is run on a server with 72 CPU cores and 180 GB of memory

'threads' option is set to 100 and all other settings are default

Full export takes 20 hours and produces 3TB of compressed text files.

Example 3:

ES backup contains 10000 indexes and 25 TB of total data in an amazon bucket, and export was run two days ago.

Tool is run on a server with 4 CPU cores and 8 GB of memory and indexNameFilter="-*beat-* -elastalert* -logstash -.*"

Partial export takes 2 hours and produces additional 13 GB of compressed text files.