FSCrawler Documentation

Release 2.5

David Pilato

Aug 05, 2018

Installation Guide

Welcome to the FS Crawler for Elasticsearch.

This crawler helps to index binary documents such as PDF, Open Office, MS Office.

Main features:

- Local file system (or a mounted drive) crawling and index new files, update existing ones and removes old ones.
- Remote file system over SSH crawling.
- REST interface to let you "upload" your binary documents to elasticsearch.

Note: FS Crawler 2.5 is using Tika 1.18 and Elasticsearch Rest Client 6.3.2.

CHAPTER 1

Download FSCrawler

You can download FSCrawler 2.5 from this link: fscrawler-2.5.

Tip: This is a **stable** version. You can choose another version than 2.5 in Maven Central.

You can also download a **SNAPSHOT** version from Sonatype.

The distribution contains:

```
$ tree
.
├── LICENSE
├── NOTICE
├── README.md
├── bin
│   ├── fscrawler
│   └── fscrawler.bat
└── lib
    ├── ... All needed jars
```

CHAPTER 2

Upgrade FSCrawler

It can happen that you need to upgrade a mapping or reindex an entire index before starting fscrawler after a version upgrade. Read carefully the following update instructions.

To update fscrawler, just download the new version, unzip it in another directory and launch it as usual. It will still pick up settings from the configuration directory. Of course, you need to stop first the existing running instances.

2.1 Upgrade to 2.2

- fscrawler comes with new default mappings for files. They have better defaults as they consume less disk space and CPU at index time. You should remove existing files in `~/.fscrawler/_default/_mappings` before starting the new version so default mappings will be updated. If you modified manually mapping files, apply the modification you made on sample files.
- `excludes` is now set by default for new jobs to `["~*"]`. In previous versions, any file or directory containing a ~ was excluded. Which means that if in your jobs, you are defining any exclusion rule, you need to add `*~*` if you want to get back the exact previous behavior.
- If you were indexing `json` or `xml` documents with the `filename_as_id` option set, we were previously removing the suffix of the file name, like indexing `1.json` was indexed as `1`. With this new version, we don't remove anymore the suffix. So the `_id` for your document will be now `1.json`.

2.2 Upgrade to 2.3

- fscrawler comes with new mapping for folders. The change is really tiny so you can skip this step if you wish. We basically removed `name` field in the folder mapping as it was unused.
- The way FSCrawler computes now `path.virtual` for docs has changed. It now includes the filename. Instead of `/path/to` you will now get `/path/to/file.txt`.
- The way FSCrawler computes now `virtual` for folders is now consistent with what you can see for folders.

- `path.encoded` in documents and `encoded` in folders have been removed as not needed by FSCrawler after all.
- *OCR integration* is now properly activated for PDF documents. This can be time, cpu and memory consuming though. You can disable explicitly it by setting `fs.pdf_ocr` to `false`.
- All dates are now indexed in elasticsearch in UTC instead of without any time zone. For example, we were indexing previously a date like `2017-05-19T13:24:47.000`. Which was producing bad results when you were located in a time zone other than UTC. It's now indexed as `2017-05-19T13:24:47.000+0000`.
- In order to be compatible with the coming 6.0 elasticsearch version, we need to get rid of types as only one type per index is still supported. Which means that we now create index named `job_name` and `job_name_folder` instead of one index `job_name` with two types `doc` and `folder`. If you are upgrading from FSCrawler 2.2, it requires that you reindex your existing data either by deleting the old index and running again FSCrawler or by using the reindex API as follows:

```
# Create folder index job_name_folder based on existing folder data
POST _reindex
{
  "source": {
    "index": "job_name",
    "type": "folder"
  },
  "dest": {
    "index": "job_name_folder"
  }
}
# Remove old folder data from job_name index
POST job_name/folder/_delete_by_query
{
  "query": {
    "match_all": {}
  }
}
```

Note that you will need first to create the right settings and mappings so you can then run the reindex job. You can do that by launching `bin/fscrawler job_name --loop 0`.

Better, you can run `bin/fscrawler job_name --upgrade` and let FSCrawler do all that for you. Note that this can take a loooong time.

Also please be aware that some APIs used by the upgrade action are only available from elasticsearch 2.3 (reindex) or elasticsearch 5.0 (delete by query). If you are running an older version than 5.0 you need first to upgrade elasticsearch.

This procedure only applies if you did not set previously `elasticsearch.type` setting (default value was `doc`). If you did, then you also need to reindex the existing documents to the default `_doc` type as per elasticsearch 6.x (or `doc` for 5.x series):

```
# Copy old type doc to the default doc type
POST _reindex
{
  "source": {
    "index": "job_name",
    "type": "your_type_here"
  },
  "dest": {
    "index": "job_name",
    "type": "_doc"
  }
```

(continues on next page)

(continued from previous page)

```
}
# Remove old type data from job_name index
POST job_name/your_type_here/_delete_by_query
{
  "query": {
    "match_all": {}
  }
}
```

But note that this last step can take a very loooong time and will generate a lot of IO on your disk. It might be easier in such case to restart fscrawler from scratch.

- As seen in the previous point, we now have 2 indices instead of a single one. Which means that `elasticsearch.index` setting has been split to `elasticsearch.index` and `elasticsearch.index_folder`. By default, it's set to the crawler name and the crawler name plus `_folder`. Note that the `upgrade` feature performs that change for you.
- fscrawler has removed now mapping files `doc.json` and `folder.json`. Mapping for doc is merged within `_settings.json` file and folder mapping is now part of `_settings_folder.json`. Which means you can remove old files to avoid confusion. You can simply remove existing files in `~/.fscrawler/_default` before starting the new version so default files will be created again.

2.3 Upgrade to 2.4

- No specific step needed. Just note that mapping changed as we support more metadata. Might be useful to run similar steps as for 2.2 upgrade.

2.4 Upgrade to 2.5

- A bug was causing a lot of data going over the wire each time FSCrawler was running. To fix this issue, we changed the default mapping and we set `store: true` on field `file.filename`. If this field is not stored and `remove_deleted` is `true` (default), FSCrawler will fail while crawling your documents. You need to create the new mapping accordingly and reindex your existing data either by deleting the old index and running again FSCrawler or by using the reindex API as follows:

```
# Backup old index data
POST _reindex
{
  "source": {
    "index": "job_name"
  },
  "dest": {
    "index": "job_name_backup"
  }
}
# Remove job_name index
DELETE job_name
```

Restart FSCrawler with the following command. It will just create the right mapping again.

```
$ bin/fscrawler job_name --loop 0
```

Then restore old data:

```
POST _reindex
{
  "source": {
    "index": "job_name_backup"
  },
  "dest": {
    "index": "job_name"
  }
}
# Remove backup index
DELETE job_name_backup
```

The default mapping changed for FSCrawler for `meta.raw.*` fields. Might be better to reindex your data.

- The `excludes` parameter is also used for directory names. But this new implementation also brings a breaking change if you were using `excludes` previously. In the previous implementation, the regular expression was only applied to the filename. It's now applied to the full virtual path name.

  For example if you have a `/tmp` dir as follows:

```
/tmp
└── folder
    ├── foo.txt
    └── bar.txt
```

  Previously excluding `foo.txt` was excluding the virtual file `/folder/foo.txt`. If you still want to exclude any file named `foo.txt` whatever its directory you now need to specify `*/foo.txt`:

```
{
  "name" : "test",
  "fs": {
    "excludes": [
      "*/foo.txt"
    ]
  }
}
```

  For more information, read *Includes and excludes*.

- For new indices, FSCrawler now uses `_doc` as the default type name for clusters running elasticsearch 6.x or superior.

CHAPTER 3

Getting Started

You need to have at least **Java 1.8.** and have properly configured `JAVA_HOME` to point to your Java installation directory. For example on MacOS you can define in your `~/.bash_profile` file:

```
export JAVA_HOME=`/usr/libexec/java_home -v 1.8`
```

3.1 Start FSCrawler

Start FSCrawler with:

```
bin/fscrawler job_name
```

FSCrawler will read a local file (default to `~/.fscrawler/{job_name}/_settings.json`). If the file does not exist, FSCrawler will propose to create your first job.

```
$ bin/fscrawler job_name
18:28:58,174 WARN  [f.p.e.c.f.FsCrawler] job [job_name] does not exist
18:28:58,177 INFO  [f.p.e.c.f.FsCrawler] Do you want to create it (Y/N)?
y
18:29:05,711 INFO  [f.p.e.c.f.FsCrawler] Settings have been created in [~/.fscrawler/
↪job_name/_settings.json]. Please review and edit before relaunch
```

Create a directory named `/tmp/es` or `c:\tmp\es`, add some files you want to index in it and start again:

```
$ bin/fscrawler --config_dir ./test job_name
18:30:34,330 INFO  [f.p.e.c.f.FsCrawlerImpl] Starting FS crawler
18:30:34,332 INFO  [f.p.e.c.f.FsCrawlerImpl] FS crawler started in watch mode. It␣
↪will run unless you stop it with CTRL+C.
18:30:34,682 INFO  [f.p.e.c.f.FsCrawlerImpl] FS crawler started for [job_name] for [/
↪tmp/es] every [15m]
```

If you did not create the directory, FSCrawler will complain until you fix it:

```
18:30:34,683 WARN  [f.p.e.c.f.FsCrawlerImpl] Error while indexing content from /tmp/
↪es: /tmp/es doesn't exists.
```

You can also run FSCrawler without arguments. It will give you the list of existing jobs and will allow you to choose one:

```
$ bin/fscrawler
18:33:00,624 INFO  [f.p.e.c.f.FsCrawler] No job specified. Here is the list of␣
↪existing jobs:
18:33:00,629 INFO  [f.p.e.c.f.FsCrawler] [1] - job_name
18:33:00,629 INFO  [f.p.e.c.f.FsCrawler] Choose your job [1-1]...
1
18:33:06,151 INFO  [f.p.e.c.f.FsCrawlerImpl] Starting FS crawler
```

3.2 Searching for docs

This is a common use case in elasticsearch, we want to search for something! ;-)

```
GET docs/doc/_search
{
  "query" : {
    "query_string": {
      "query": "I am searching for something !"
    }
  }
}
```

See *Search examples* for more examples.

CHAPTER 4

Crawler options

By default, FSCrawler will read your file from `/tmp/es` every 15 minutes. You can change those settings by modifying `~/.fscrawler/{job_name}/_settings.json` file where `{job_name}` is the name of the job you just created.

```
{
  "name" : "job_name",
  "fs" : {
    "url" : "/path/to/data/dir",
    "update_rate" : "15m"
  }
}
```

You can change also `update_rate` to watch more or less frequently for changes.

If you just want FSCrawler to run once and exit, run it with `--loop` option:

```
$ bin/fscrawler job_name --loop 1
18:47:37,487 INFO  [f.p.e.c.f.FsCrawlerImpl] Starting FS crawler
18:47:37,854 INFO  [f.p.e.c.f.FsCrawlerImpl] FS crawler started for [job_name] for [/
↪tmp/es] every [15m]
...
18:47:37,855 INFO  [f.p.e.c.f.FsCrawlerImpl] FS crawler is stopping after 1 run
18:47:37,959 INFO  [f.p.e.c.f.FsCrawlerImpl] FS crawler [job_name] stopped
```

If you have already ran FSCrawler and want to restart (which means reindex existing documents), use the `--restart` option:

```
$ bin/fscrawler job_name --loop 1 --restart
```

You will find more information about settings in the following sections:

- *CLI options*
- *Local FS settings*
- *SSH settings*

- *Elasticsearch settings*

CHAPTER 5

Starting with a REST gateway

New in version 2.2.

FSCrawler can be a nice gateway to elasticsearch if you want to upload binary documents and index them into elasticsearch without writing by yourself all the code to extract data and communicate with elasticsearch.

To start FSCrawler with the REST service, use the `--rest` option. A good idea is also to combine it with `--loop 0` so you won't index local files but only listen to incoming REST requests:

```
$ bin/fscrawler job_name --loop 0 --rest
18:55:37,851 INFO  [f.p.e.c.f.FsCrawlerImpl] Starting FS Crawler
18:55:39,237 INFO  [f.p.e.c.f.FsCrawlerImpl] FS crawler Rest service started on␣
↪[http://127.0.0.1:8080/fscrawler]
```

Check the service is working with:

```
curl http://127.0.0.1:8080/fscrawler/
```

It will give you back a JSON document.

The you can start uploading your binary files:

```
echo "This is my text" > test.txt
curl -F "file=@test.txt" "http://127.0.0.1:8080/fscrawler/_upload"
```

It will index the file into elasticsearch and will give you back the elasticsearch URL for the created document, like:

```
{
  "ok" : true,
  "filename" : "test.txt",
  "url" : "http://127.0.0.1:9200/fscrawler-rest-tests_doc/doc/
↪dd18bf3a8ea2a3e53e2661c7fb53534"
}
```

Read the *REST service* chapter for more information.

CHAPTER 6

Supported formats

FSCrawler supports all formats Tika supports, like:

- HTML
- Microsoft Office
- Open Office
- PDF
- Images
- MP3
- ...

CHAPTER 7

Tips and tricks

7.1 Moving files to a "watched" directory

When moving an existing file to the directory FSCrawler is watching, you need to explicitly `touch` all the files as when moved, the files are keeping their original date intact:

```
# single file
touch file_you_moved

# all files
find  -type f  -exec touch {} +

# all .txt files
find  -type f  -name "*.txt" -exec touch {} +
```

Or you need to *restart* from the beginning with the `--restart` option which will reindex everything.

7.2 Indexing from HDFS drive

There is no specific support for HDFS in FSCrawler. But you can mount your HDFS on your machine and run FS crawler on this mount point. You can also read details about HDFS NFS Gateway.

7.3 OCR integration

New in version 2.3.

To deal with images containing text, just install Tesseract. Tesseract will be auto-detected by Tika or you can explicitly set the path to tesseract binary. Then add an image (png, jpg, ...) into your Fscrawler *Root directory*. After the next index update, the text will be indexed and placed in "_source.content".

By default, FSCrawler will try to extract also images from your PDF documents and run OCR on them. This can be a CPU intensive operation. If you don't mean to run OCR on PDF but only on images, you can set `fs.pdf_ocr` to `false`:

```
{
  "name" : "test",
  "fs" : {
    "pdf_ocr" : false
  }
}
```

7.3.1 OCR settings

Here is a list of OCR settings (under `fs.ocr` prefix)':

Name	Default value	Documentation
`fs.ocr.language`	`"eng"`	*OCR Language*
`fs.ocr.path`	`null`	*OCR Path*
`fs.ocr.data_path`	`null`	*OCR Data Path*
`fs.ocr.output_type`	`txt`	*OCR Output Type*

7.3.2 OCR Language

If you have installed a Tesseract Language pack, you can use it when parsing your documents by setting `fs.ocr.language` property in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir",
    "ocr" : {
      "language": "eng"
    }
  }
}
```

7.3.3 OCR Path

If your Tesseract application is not available in default system PATH, you can define the path to use by setting `fs.ocr.path` property in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir",
    "ocr" : {
      "path": "/path/to/tesseract/executable"
    }
  }
}
```

When you set it, it's highly recommended to set the *OCR Data Path*.

7.3.4 OCR Data Path

Set the path to the 'tessdata' folder, which contains language files and config files if Tesseract can not be automatically detected. You can define the path to use by setting `fs.ocr.data_path` property in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir",
    "ocr" : {
      "path": "/path/to/tesseract/executable",
      "data_path": "/path/to/tesseract/tessdata"
    }
  }
}
```

7.3.5 OCR Output Type

New in version 2.5.

Set the output type from ocr process. `fs.ocr.output_type` property can be defined to `txt` or `hocr` in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir",
    "ocr" : {
      "output_type": "hocr"
    }
  }
}
```

Note: When omitted, `txt` value is used.

7.4 Using docker

To use FSCrawler with docker, check docker-fscrawler recipe.

CHAPTER 8

Status files

Once the crawler is running, it will write status information and statistics in:

- `~/.fscrawler/{job_name}/_settings.json`
- `~/.fscrawler/{job_name}/_status.json`

It means that if you stop the job at some point, FSCrawler will restart it from where it stops.

CHAPTER 9

CLI options

- `--help` displays help
- `--silent` runs in silent mode. No output is generated.
- `--debug` runs in debug mode.
- `--trace` runs in trace mode (more verbose than debug).
- `--config_dir` defines directory where jobs are stored instead of default `~/.fscrawler`.
- `--username` defines the username to use when using an secured version of elasticsearch cluster. Read *Using Credentials (X-Pack)*.
- `--upgrade` runs a reindex operation for indices created with an older version. See *Upgrade*.
- `--loop x` defines the number of runs we want before exiting. See *Loop*.
- `--restart` restart a job from scratch. See *Restart*.
- `--rest` starts the REST service. See *Rest*.

9.1 Upgrade

`--upgrade` runs a reindex operation for indices created with an older version which was using multiple types within the same index. More on this in *Upgrade to 2.3* section.

9.2 Loop

New in version 2.2.

`--loop x` defines the number of runs we want before exiting:

- `X` where X is a negative value means infinite, like `-1` (default)
- `0` means that we don't run any crawling job (useful when used with rest).

- `X` where X is a positive value is the number of runs before it stops.

If you want to scan your hard drive only once, run with `--loop 1`.

9.3 Restart

New in version 2.2.

You can tell FSCrawler that it must restart from the beginning by using `--restart` option:

```
bin/fscrawler job_name --restart
```

In that case, the `{job_name}/_status.json` file will be removed.

9.4 Rest

New in version 2.3.

If you want to run the *REST service* without scanning your hard drive, launch with:

```
bin/fscrawler --rest --loop 0
```

CHAPTER 10

JVM Settings

If you want to provide JVM settings, like defining memory allocated to FSCrawler, you can define a system property named `FS_JAVA_OPTS`:

```
FS_JAVA_OPTS="-Xmx521m -Xms521m" bin/fscrawler
```

CHAPTER 11

Configuring an external logger configuration file

If you want to define an external `log4j2.xml` file, you can use the `log4j.configurationFile` JVM parameter which you can define in `FS_JAVA_OPTS` variable if you wish:

```
FS_JAVA_OPTS="-Dlog4j.configurationFile=path/to/log4j2.xml" bin/fscrawler
```

You can use the default log4j2.xml file as an example to start with.

CHAPTER 12

Job file specification

The job file must comply to the following `json` specifications:

```
{
  "name" : "job_name",
  "fs" : {
    "url" : "/path/to/docs",
    "update_rate" : "5m",
    "includes" : [ "*.doc", "*.xls" ],
    "excludes" : [ "resume.doc" ],
    "json_support" : false,
    "filename_as_id" : true,
    "add_filesize" : true,
    "remove_deleted" : true,
    "add_as_inner_object" : false,
    "store_source" : true,
    "index_content" : true,
    "indexed_chars" : "10000.0",
    "attributes_support" : false,
    "raw_metadata" : true,
    "xml_support" : false,
    "index_folders" : true,
    "lang_detect" : false,
    "continue_on_error" : false,
    "pdf_ocr" : true,
    "ocr" : {
      "language" : "eng",
      "path": "/path/to/tesseract/if/not/available/in/PATH",
      "data_path": "/path/to/tesseract/tessdata/if/needed"
    }
  },
  "server" : {
    "hostname" : "localhost",
    "port" : 22,
    "username" : "dadoonet",
```

(continues on next page)

(continued from previous page)

```
    "password" : "password",
    "protocol" : "SSH",
    "pem_path" : "/path/to/pemfile"
  },
  "elasticsearch" : {
    "nodes" : [ {
      // With Cloud ID
      "cloud_id" : "CLOUD_ID"
    }, {
      // With scheme, host and port
      "host" : "127.0.0.1",
      "port" : 9200,
      "scheme" : "HTTP"
    } ],
    "index" : "docs",
    "bulk_size" : 1000,
    "flush_interval" : "5s",
    "byte_size" : "10mb",
    "username" : "elastic",
    "password" : "password"
  },
  "rest" : {
    "scheme" : "HTTP",
    "host" : "127.0.0.1",
    "port" : 8080,
    "endpoint" : "fscrawler"
  }
}
```

Here is a list of existing top level settings:

Name	Documentation
`name` (mandatory field)	*The most simple crawler*
`fs`	*Local FS settings*
`elasticsearch`	*Elasticsearch settings*
`server`	*SSH settings*
`rest`	*REST service*

CHAPTER 13

The most simple crawler

You can define the most simple crawler job by writing a `~/.fscrawler/test/_settings.json` file as follow:

```
{
  "name" : "test"
}
```

This will scan every 15 minutes all documents available in `/tmp/es` dir and will index them into `test_doc` index. It will connect to an elasticsearch cluster running on `127.0.0.1`, port `9200`.

Note: `name` is a mandatory field.

CHAPTER 14

Local FS settings

Here is a list of Local FS settings (under `fs.` prefix)‘:

Name	Default value	Documentation
`fs.url`	`"/tmp/es"`	*Root directory*
`fs.update_rate`	`"15m"`	*Update Rate*
`fs.includes`	`null`	*Includes and excludes*
`fs.excludes`	`["~*"]`	*Includes and excludes*
`fs.filters`	`null`	*Filter content*
`fs.json_support`	`false`	*Indexing JSon docs*
`fs.xml_support`	`false`	*Indexing XML docs*
`fs.add_as_inner_object`	`false`	*Add as Inner Object*
`fs.index_folders`	`true`	*Index folders*
`fs.attributes_support`	`false`	*Adding file attributes*
`fs.raw_metadata`	`true`	*Disabling raw metadata*
`fs.filename_as_id`	`false`	*Using filename as elasticsearch _id*
`fs.add_filesize`	`true`	*Disabling file size field*
`fs.remove_deleted`	`true`	*Ignore deleted files*
`fs.store_source`	`false`	*Storing binary source document*
`fs.index_content`	`true`	*Ignore content*
`fs.lang_detect`	`false`	*Language detection*
`fs.continue_on_error`	`false`	*Continue on Error*
`fs.pdf_ocr`	`true`	*OCR integration*
`fs.indexed_chars`	`100000.0`	*Extracted characters*
`fs.ignore_above`	`null`	*Ignore above*
`fs.checksum`	`null`	*File Checksum*

14.1 Root directory

Define `fs.url` property in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir"
  }
}
```

For Windows users, use a form like `c:/tmp` or `c:\\tmp`.

14.2 Update rate

By default, `update_rate` is set to `15m`. You can modify this value using any compatible time unit.

For example, here is a 15 minutes update rate:

```
{
  "name": "test",
  "fs": {
    "update_rate": "15m"
  }
}
```

Or a 3 hours update rate:

```
{
  "name": "test",
  "fs": {
    "update_rate": "3h"
  }
}
```

`update_rate` is the pause duration between the last time we read the file system and another run. Which means that if you set it to `15m`, the next scan will happen on 15 minutes after the end of the current scan, whatever its duration.

14.3 Includes and excludes

Let's say you want to index only docs like `*.doc` and `*.pdf` but `resume*`. So `resume_david.pdf` won't be indexed.

Define `fs.includes` and `fs.excludes` properties in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs": {
    "includes": [
      "*/*.doc",
      "*/*.pdf"
    ],
    "excludes": [
      "*/resume*"
    ]
  }
}
```

By default, FSCrawler will exclude files starting with `~`.

New in version 2.5.

It also applies to directory names. So if you want to ignore `.ignore` dir, just add `.ignore` as an excluded name. Note that `includes` and `excludes` apply to directory names as well.

Let's take the following example with the `root` dir as `/tmp`:

```
/tmp
├── folderA
│   ├── subfolderA
│   ├── subfolderB
│   └── subfolderC
├── folderB
│   ├── subfolderA
│   ├── subfolderB
│   └── subfolderC
└── folderC
    ├── subfolderA
    ├── subfolderB
    └── subfolderC
```

If you define the following `fs.excludes` property in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs": {
    "excludes": [
      "/folderB/subfolder*"
    ]
  }
}
```

Then all files but the ones in `/folderB/subfolderA`, `/folderB/subfolderB` and `/folderB/subfolderC` will be indexed.

14.4 Filter content

New in version 2.5.

You can filter out documents you would like to index by adding one or more regular expression that match the extracted content. Documents which are not matching will be simply ignored and not indexed.

If you define the following `fs.filters` property in your `~/.fscrawler/test/_settings.json` file:

```
{
  "name" : "test",
  "fs": {
    "filters": [
      ".*foo.*",
      "^4\\d{3}([\\ \\-]?)\\d{4}\\1\\d{4}\\1\\d{4}$"
    ]
  }
}
```

With this example, only documents which contains the word `foo` and a VISA credit card number with the form like `4012888888881881`, `4012 8888 8888 1881` or `4012-8888-8888-1881` will be indexed.

14.5 Indexing JSon docs

If you want to index JSon files directly without parsing with Tika, you can set `json_support` to `true`. JSon contents will be stored directly under _source. If you need to keep JSon documents synchronized to the index, set option *Add as Inner Object* which stores additional metadata and the JSon contents under field `object`.

```
{
  "name" : "test",
  "fs" : {
    "json_support" : true
  }
}
```

Of course, if you did not define a mapping before launching the crawler, Elasticsearch will auto guess the mapping.

14.6 Indexing XML docs

New in version 2.2.

If you want to index XML files and convert them to JSON, you can set `xml_support` to `true`. The content of XML files will be added directly under _source. If you need to keep XML documents synchronized to the index, set option *Add as Inner Object* which stores additional metadata and the XML contents under field `object`.

```
{
  "name" : "test",
  "fs" : {
    "xml_support" : true
  }
}
```

Of course, if you did not define a mapping before launching the crawler, Elasticsearch will auto guess the mapping.

14.7 Add as Inner Object

The default settings store the contents of json and xml documents directly onto the _source element of elasticsearch documents. Thereby, there is no metadata about file and path settings, which are necessary to determine if a document is deleted or updated. New files will however be added to the index, (determined by the file timestamp).

If you need to keep json or xml documents synchronized to elasticsearch, you should set this option.

```
{
  "name" : "test",
  "fs" : {
    "add_as_inner_object" : true
  }
}
```

14.8 Index folders

New in version 2.2.

By default FSCrawler will index folder names in the folder index. If you don't want to index those folders, you can set `index_folders` to `false`.

Note that in that case, FSCrawler won't be able to detect removed folders so any document has been indexed in elasticsearch, it won't be removed when you remove or move the folder away.

```
{
  "name" : "test",
  "fs" : {
    "index_folders" : false
  }
}
```

14.9 Dealing with multiple types and multiple dirs

If you have more than one type, create as many crawlers as types:

`~/.fscrawler/test_type1/_settings.json`:

```
{
  "name": "test_type1",
  "fs": {
    "url": "/tmp/type1",
    "json_support" : true
  },
  "elasticsearch": {
    "index": "mydocs1",
    "index_folder": "myfolders1"
  }
}
```

`~/.fscrawler/test_type2/_settings.json`:

```
{
  "name": "test_type2",
  "fs": {
    "url": "/tmp/type2",
    "json_support" : true
  },
  "elasticsearch": {
    "index": "mydocs2",
    "index_folder": "myfolders2"
  }
}
```

`~/.fscrawler/test_type3/_settings.json`:

```
{
  "name": "test_type3",
  "fs": {
    "url": "/tmp/type3",
    "xml_support" : true
  },
  "elasticsearch": {
    "index": "mydocs3",
    "index_folder": "myfolders3"
```

(continues on next page)

(continued from previous page)

```
  }
}
```

14.10 Dealing with multiple types within the same dir

You can also index many types from one single dir using two crawlers scanning the same dir and by setting `includes` parameter:

`~/.fscrawler/test_type1.json`:

```
{
  "name": "test_type1",
  "fs": {
    "url": "/tmp",
    "includes": [ "type1*.json" ],
    "json_support" : true
  },
  "elasticsearch": {
    "index": "mydocs1",
    "index_folder": "myfolders1"
  }
}
```

`~/.fscrawler/test_type2.json`:

```
{
  "name": "test_type2",
  "fs": {
    "url": "/tmp",
    "includes": [ "type2*.json" ],
    "json_support" : true
  },
  "elasticsearch": {
    "index": "mydocs2",
    "index_folder": "myfolders2"
  }
}
```

`~/.fscrawler/test_type3.json`:

```
{
  "name": "test_type3",
  "fs": {
    "url": "/tmp",
    "includes": [ "*.xml" ],
    "xml_support" : true
  },
  "elasticsearch": {
    "index": "mydocs3",
    "index_folder": "myfolders3"
  }
}
```

14.11 Using filename as elasticsearch _id

Please note that the document _id is always generated (hash value) from the filename to avoid issues with special characters in filename. You can force to use the _id to be the filename using filename_as_id attribute:

```
{
  "name" : "test",
  "fs" : {
    "filename_as_id" : true
  }
}
```

14.12 Adding file attributes

If you want to add file attributes such as attributes.owner, attributes.group and attributes.permissions, you can set attributes_support to true.

```
{
  "name" : "test",
  "fs" : {
    "attributes_support" : true
  }
}
```

Note: On Windows systems, attributes.group and attributes.permissions are not generated.

14.13 Disabling raw metadata

By default, FSCrawler will extract all found metadata within meta.raw object. If you want to disable this feature, you can set raw_metadata to false.

```
{
  "name" : "test",
  "fs" : {
    "raw_metadata" : false
  }
}
```

Generated raw metadata depends on the file format itself.

For example, a PDF document could generate:

```
{
   "date" : "2016-07-07T08:37:42Z",
   "pdf:PDFVersion" : "1.5",
   "xmp:CreatorTool" : "Microsoft Word",
   "Keywords" : "keyword1, keyword2",
   "access_permission:modify_annotations" : "true",
   "access_permission:can_print_degraded" : "true",
   "subject" : "Test Tika Object",
```

(continues on next page)

(continued from previous page)


```
   "dc:creator" : "David Pilato",
   "dcterms:created" : "2016-07-07T08:37:42Z",
   "Last-Modified" : "2016-07-07T08:37:42Z",
   "dcterms:modified" : "2016-07-07T08:37:42Z",
   "dc:format" : "application/pdf; version=1.5",
   "title" : "Test Tika title",
   "Last-Save-Date" : "2016-07-07T08:37:42Z",
   "access_permission:fill_in_form" : "true",
   "meta:save-date" : "2016-07-07T08:37:42Z",
   "pdf:encrypted" : "false",
   "dc:title" : "Test Tika title",
   "modified" : "2016-07-07T08:37:42Z",
   "cp:subject" : "Test Tika Object",
   "Content-Type" : "application/pdf",
   "X-Parsed-By" : "org.apache.tika.parser.DefaultParser",
   "creator" : "David Pilato",
   "meta:author" : "David Pilato",
   "dc:subject" : "keyword1, keyword2",
   "meta:creation-date" : "2016-07-07T08:37:42Z",
   "created" : "Thu Jul 07 10:37:42 CEST 2016",
   "access_permission:extract_for_accessibility" : "true",
   "access_permission:assemble_document" : "true",
   "xmpTPg:NPages" : "2",
   "Creation-Date" : "2016-07-07T08:37:42Z",
   "access_permission:extract_content" : "true",
   "access_permission:can_print" : "true",
   "meta:keyword" : "keyword1, keyword2",
   "Author" : "David Pilato",
   "access_permission:can_modify" : "true"
}
```


Where a MP3 file would generate:


```
{
   "xmpDM:genre" : "Vocal",
   "X-Parsed-By" : "org.apache.tika.parser.DefaultParser",
   "creator" : "David Pilato",
   "xmpDM:album" : "FS Crawler",
   "xmpDM:trackNumber" : "1",
   "xmpDM:releaseDate" : "2016",
   "meta:author" : "David Pilato",
   "xmpDM:artist" : "David Pilato",
   "dc:creator" : "David Pilato",
   "xmpDM:audioCompressor" : "MP3",
   "title" : "Test Tika",
   "xmpDM:audioChannelType" : "Stereo",
   "version" : "MPEG 3 Layer III Version 1",
   "xmpDM:logComment" : "Hello but reverted",
   "xmpDM:audioSampleRate" : "44100",
   "channels" : "2",
   "dc:title" : "Test Tika",
   "Author" : "David Pilato",
   "xmpDM:duration" : "1018.775146484375",
   "Content-Type" : "audio/mpeg",
   "samplerate" : "44100"
}
```

Note: All fields are generated as text even though they can be valid booleans or numbers.

The `meta.raw.*` fields have a default mapping applied:

```
{
  "type": "text",
  "fields": {
    "keyword": {
      "type": "keyword",
      "ignore_above": 256
    }
  }
}
```

If you want specifically tell elasticsearch to use a date type or a numeric type for some fields, you need to modify the default template provided by FSCrawler.

Note: Note that dots in metadata names will be replaced by a `:`. For example `PTEX.Fullbanner` will be indexed as `PTEX:Fullbanner`.

14.14 Disabling file size field

By default, FSCrawler will create a field to store the original file size in octets. You can disable it using 'add_filesize' option:

```
{
  "name" : "test",
  "fs" : {
    "add_filesize" : false
  }
}
```

14.15 Ignore deleted files

If you don't want to remove indexed documents when you remove a file or a directory, you can set `remove_deleted` to `false` (default to `true`):

```
{
  "name" : "test",
  "fs" : {
    "remove_deleted" : false
  }
}
```

14.16 Ignore content

If you don't want to extract file content but only index filesystem metadata such as filename, date, size and path, you can set `index_content` to `false` (default to `true`):

```
{
  "name" : "test",
  "fs" : {
    "index_content" : false
  }
}
```

14.17 Continue on Error

New in version 2.3.

By default FSCrawler will immediately stop indexing if he hits a Permission denied exception. If you want to just skip this File and continue with the rest of the directory tree you can set `continue_on_error` to `true` (default to `false`):

```
{
  "name" : "test",
  "fs" : {
    "continue_on_error" : true
  }
}
```

14.18 Language detection

New in version 2.2.

You can ask for language detection using `lang_detect` option:

```
{
  "name" : "test",
  "fs" : {
    "lang_detect" : true
  }
}
```

In that case, a new field named `meta.language` is added to the generated JSon document.

If you are using elasticsearch 5.0 or superior, you can use this value to send your document to a specific index using a *Node Ingest pipeline*.

For example, you can define a pipeline named `langdetect` with:

```
PUT _ingest/pipeline/langdetect
{
  "description" : "langdetect pipeline",
  "processors" : [
    {
      "set": {
        "field": "_index",
        "value": "myindex-{{meta.language}}"
      }
    }
  ]
}
```

In FSCrawler settings, set both `fs.lang_detect` and `elasticsearch.pipeline` options:

```
{
  "name" : "test",
  "fs" : {
    "lang_detect" : true
  },
  "elasticsearch" : {
    "pipeline" : "langdetect"
  }
}
```

And then, a document containing french text will be sent to `myindex-fr`. A document containing english text will be sent to `myindex-en`.

You can also imagine changing the field name from `content` to `content-fr` or `content-en`. That will help you to define the correct analyzer to use.

Language detection might detect more than one language in a given text but only the most accurate will be set. Which means that if you have a document containing 80% of french and 20% of english, the document will be marked as `fr`.

Note that language detection is CPU and time consuming.

14.19 Storing binary source document

You can store in elasticsearch itself the binary document (BASE64 encoded) using `store_source` option:

```
{
  "name" : "test",
  "fs" : {
    "store_source" : true
  }
}
```

In that case, a new field named `attachment` is added to the generated JSon document. This field is not indexed. Default mapping for `attachment` field is:

```
{
  "_doc" : {
    "properties" : {
      "attachment" : {
        "type" : "binary",
        "doc_values" : false
      }
      // ... Other properties here
    }
  }
}
```

14.20 Extracted characters

By default FSCrawler will extract only the first 100 000 characters. But, you can set `indexed_chars` to `5000` in FSCrawler settings in order to overwrite this default settings.

```
{
  "name": "test",
  "fs": {
    "indexed_chars": "5000"
  }
}
```

This number can be either a fixed size, number of characters that is, or a percent using `%` sign. The percentage value will be applied to the filesize to determine the number of character the crawler needs to extract.

If you want to index only `80%` of filesize, define `indexed_chars` to `"80%"`. Of course, if you want to index the full document, you can set this property to `"100%"`. Double values are also supported so `"0.01%"` is also a correct value.

Compressed files: If your file is compressed, you might need to increase `indexed_chars` to more than `"100%"`. For example, `"150%"`.

If you want to extract the full content, define `indexed_chars` to `"-1"`.

Note: Tika requires to allocate in memory a data structure to extract text. Setting `indexed_chars` to a high number will require more memory!

14.21 Ignore Above

New in version 2.5.

By default FSCrawler will send to Tika every single file, whatever its size. But some files on your file system might be a way too big to be parsed.

Set `ignore_above` to the desired value of the limit.

```
{
  "name": "test",
  "fs": {
    "ignore_above": "5mb"
  }
}
```

14.22 File checksum

If you want FSCrawler to generate a checksum for each file, set `checksum` to the algorithm you wish to use to compute the checksum, such as `MD5` or `SHA-1`.

```
{
  "name": "test",
  "fs": {
    "checksum": "MD5"
  }
}
```

CHAPTER 15

SSH settings

You can index files remotely using SSH.

Here is a list of SSH settings (under `server.` prefix)‘:

Name	Default value	Documentation
`server.hostname`	`null`	Hostname
`server.port`	`22`	Port
`server.username`	`null`	*Username / Password*
`server.password`	`null`	*Username / Password*
`server.protocol`	`"local"`	Set it to `ssh`
`server.pem_path`	`null`	*Using Username / PEM file*

15.1 Username / Password

Let’s say you want to index from a remote server using SSH:

- FS URL: `/path/to/data/dir/on/server`
- Server: `mynode.mydomain.com`
- Username: `username`
- Password: `password`
- Protocol: `ssh` (default to `local`)
- Port: `22` (default to `22`)

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir/on/server"
  },
```

(continues on next page)

(continued from previous page)

```
  "server" : {
    "hostname" : "mynode.mydomain.com",
    "port" : 22,
    "username" : "username",
    "password" : "password",
    "protocol" : "ssh"
  }
}
```

15.2 Using Username / PEM file

Let's say you want to index from a remote server using SSH:

- FS URL: `/path/to/data/dir/on/server`
- Server: `mynode.mydomain.com`
- Username: `username`
- PEM File: `/path/to/private_key.pem`
- Protocol: `ssh` (default to `local`)
- Port: `22` (default to `22`)

```
{
  "name" : "test",
  "fs" : {
    "url" : "/path/to/data/dir/on/server"
  },
  "server" : {
    "hostname" : "mynode.mydomain.com",
    "port" : 22,
    "username" : "username",
    "protocol" : "ssh",
    "pem_path": "/path/to/private_key.pem"
  }
}
```

CHAPTER 16

Elasticsearch settings

Here is a list of Elasticsearch settings (under `elasticsearch.` prefix)‘:

Name	Default value	Documentation
`elasticsearch.index`	job name	*Index settings for documents*
`elasticsearch.index_folder`	job name + `_folder`	*Index settings for folders*
`elasticsearch.bulk_size`	`100`	*Bulk settings*
`elasticsearch.flush_interval`	`"5s"`	*Bulk settings*
`elasticsearch.byte_size`	`"10mb"`	*Bulk settings*
`elasticsearch.pipeline`	`null`	*Using Ingest Node Pipeline*
`elasticsearch.nodes`	`http://127.0.0.1:9200`	*Node settings*
`elasticsearch.username`	`null`	*Using Credentials (X-Pack)*
`elasticsearch.password`	`null`	*Using Credentials (X-Pack)*

16.1 Index settings

16.1.1 Index settings for documents

By default, FSCrawler will index your data in an index which name is the same as the crawler name (`name` property) plus `_doc` suffix, like `test_doc`. You can change it by setting `index` field:

```
{
  "name" : "test",
  "elasticsearch" : {
    "index" : "docs"
  }
}
```

16.1.2 Index settings for folders

FSCrawler will also index folders in an index which name is the same as the crawler name (`name` property) plus `_folder` suffix, like `test_folder`. You can change it by setting `index_folder` field:

```
{
 "name" : "test",
 "elasticsearch" : {
   "index_folder" : "folders"
 }
}
```

16.1.3 Mappings

When FSCrawler needs to create the doc index, it applies some default settings and mappings which are read from `~/.fscrawler/_default/6/_settings.json`. You can read its content from the source.

Settings define an analyzer named `fscrawler_path` which uses a path hierarchy tokenizer.

FSCrawler applies as well a mapping automatically for the folders which can also be read from the source.

You can also display the index mapping being used with Kibana:

```
GET docs/_mapping
GET docs_folder/_mapping
```

Or fall back to the command line:

```
curl 'http://localhost:9200/docs/_mapping?pretty'
curl 'http://localhost:9200/docs_folder/_mapping?pretty'
```

Note: FSCrawler is actually applying default index settings depending on the elasticsearch version it is connected to. The default settings definitions are stored in `~/.fscrawler/_default/_mappings`:

- `2/_settings.json`: for elasticsearch 2.x series document index settings
- `2/_settings_folder.json`: for elasticsearch 2.x series folder index settings
- `5/_settings.json`: for elasticsearch 5.x series document index settings
- `5/_settings_folder.json`: for elasticsearch 5.x series folder index settings
- `6/_settings.json`: for elasticsearch 6.x series document index settings
- `6/_settings_folder.json`: for elasticsearch 6.x series folder index settings

Note: For versions before 6.x series, the type of the document is `doc`. From 6.x, the type of the document is `_doc`.

Creating your own mapping (analyzers)

If you want to define your own index settings and mapping to set analyzers for example, you can either create the index and push the mapping or define a `~/.fscrawler/_default/6/_settings.json` document which contains the index settings and mappings you wish **before starting the FSCrawler**.

The following example uses a `french` analyzer to index the `content` field.

```
{
  "settings": {
    "index.mapping.total_fields.limit": 2000,
    "analysis": {
      "analyzer": {
        "fscrawler_path": {
          "tokenizer": "fscrawler_path"
        }
      },
      "tokenizer": {
        "fscrawler_path": {
          "type": "path_hierarchy"
        }
      }
    }
  },
  "mappings": {
    "_doc": {
      "dynamic_templates": [
        {
          "raw_as_text": {
            "path_match": "meta.raw.*",
            "mapping": {
              "type": "text",
              "fields": {
                "keyword": {
                  "type": "keyword",
                  "ignore_above": 256
                }
              }
            }
          }
        }
      ],
      "properties": {
        "attachment": {
          "type": "binary",
          "doc_values": false
        },
        "attributes": {
          "properties": {
            "group": {
              "type": "keyword"
            },
            "owner": {
              "type": "keyword"
            }
          }
        },
        "content": {
          "type": "text",
          "analyzer": "french"
        },
        "file": {
          "properties": {
            "content_type": {
              "type": "keyword"
```

(continues on next page)

(continued from previous page)

```
            },
            "filename": {
              "type": "keyword",
              "store": true
            },
            "extension": {
              "type": "keyword"
            },
            "filesize": {
              "type": "long"
            },
            "indexed_chars": {
              "type": "long"
            },
            "indexing_date": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "created": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "last_modified": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "last_accessed": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "checksum": {
              "type": "keyword"
            },
            "url": {
              "type": "keyword",
              "index": false
            }
          }
        },
        "meta": {
          "properties": {
            "author": {
              "type": "text"
            },
            "date": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "keywords": {
              "type": "text"
            },
            "title": {
              "type": "text"
            },
            "language": {
              "type": "keyword"
            },
```

(continues on next page)

(continued from previous page)

```
            "format": {
              "type": "text"
            },
            "identifier": {
              "type": "text"
            },
            "contributor": {
              "type": "text"
            },
            "coverage": {
              "type": "text"
            },
            "modifier": {
              "type": "text"
            },
            "creator_tool": {
              "type": "keyword"
            },
            "publisher": {
              "type": "text"
            },
            "relation": {
              "type": "text"
            },
            "rights": {
              "type": "text"
            },
            "source": {
              "type": "text"
            },
            "type": {
              "type": "text"
            },
            "description": {
              "type": "text"
            },
            "created": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "print_date": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "metadata_date": {
              "type": "date",
              "format": "dateOptionalTime"
            },
            "latitude": {
              "type": "text"
            },
            "longitude": {
              "type": "text"
            },
            "altitude": {
              "type": "text"
            },
```

(continues on next page)

(continued from previous page)

```
            "rating": {
              "type": "byte"
            },
            "comments": {
              "type": "text"
            }
          }
        },
        "path": {
          "properties": {
            "real": {
              "type": "keyword",
              "fields": {
                "tree": {
                  "type": "text",
                  "analyzer": "fscrawler_path",
                  "fielddata": true
                },
                "fulltext": {
                  "type": "text"
                }
              }
            },
            "root": {
              "type": "keyword"
            },
            "virtual": {
              "type": "keyword",
              "fields": {
                "tree": {
                  "type": "text",
                  "analyzer": "fscrawler_path",
                  "fielddata": true
                },
                "fulltext": {
                  "type": "text"
                }
              }
            }
          }
        }
      }
    }
  }
}
```

Note that if you want to push manually the mapping to elasticsearch you can use the classic REST calls:

```
# Create index (don't forget to add the fscrawler_path analyzer)
PUT docs
{
  // Same index settings as previously seen
}
```

Define explicit mapping/settings per job

Let's say you created a job named `job_name` and you are sending documents against an elasticsearch cluster running version `6.x`.

If you create the following files, they will be picked up at job start time instead of the *default ones*:

- `~/.fscrawler/{job_name}/_mappings/6/_settings.json`
- `~/.fscrawler/{job_name}/_mappings/6/_settings_folder.json`

Tip: You can do the same for other elasticsearch versions with:

- `~/.fscrawler/{job_name}/_mappings/2/_settings.json` for 2.x series (deprecated)
- `~/.fscrawler/{job_name}/_mappings/2/_settings_folder.json` for 2.x series (deprecated)
- `~/.fscrawler/{job_name}/_mappings/5/_settings.json` for 5.x series
- `~/.fscrawler/{job_name}/_mappings/5/_settings_folder.json` for 5.x series

Replace existing mapping

Unfortunately you can not change the mapping on existing data. Therefore, you'll need first to remove existing index, which means remove all existing data, and then restart FSCrawler with the new mapping.

You might to try elasticsearch Reindex API though.

16.2 Bulk settings

FSCrawler is using bulks to send data to elasticsearch. By default the bulk is executed every 100 operations or every 5 seconds or every 10 megabytes. You can change default settings using `bulk_size`, `byte_size` and `flush_interval`:

```
{
  "name" : "test",
  "elasticsearch" : {
    "bulk_size" : 1000,
    "byte_size" : "500kb",
    "flush_interval" : "2s"
  }
}
```

Tip: Elasticsearch has a default limit of `100mb` per HTTP request as per elasticsearch HTTP Module documentation.

Which means that if you are indexing a massive bulk of documents, you might hit that limit and FSCrawler will throw an error like `entity content is too long [xxx] for the configured buffer limit [104857600]`.

You can either change this limit on elasticsearch side by setting `http.max_content_length` to a higher value but please be aware that this will consume much more memory on elasticsearch side.

Or you can decrease the `bulk_size` or `byte_size` setting to a smaller value.

16.3 Using Ingest Node Pipeline

New in version 2.2.

If you are using an elasticsearch cluster running a 5.0 or superior version, you can use an Ingest Node pipeline to transform documents sent by FSCrawler before they are actually indexed.

For example, if you have the following pipeline:

```
PUT _ingest/pipeline/fscrawler
{
  "description" : "fscrawler pipeline",
  "processors" : [
    {
      "set" : {
        "field": "foo",
        "value": "bar"
      }
    }
  ]
}
```

In FSCrawler settings, set the `elasticsearch.pipeline` option:

```
{
  "name" : "test",
  "elasticsearch" : {
    "pipeline" : "fscrawler"
  }
}
```

Note: Folder objects are not sent through the pipeline as they are more internal objects.

16.4 Node settings

FSCrawler is using elasticsearch REST layer to send data to your running cluster. By default, it connects to `127.0.0.1` on port `9200` which are the default settings when running a local node on your machine.

Of course, in production, you would probably change this and connect to a production cluster:

```
{
  "name" : "test",
  "elasticsearch" : {
    "nodes" : [
      { "host" : "mynode1.mycompany.com", "port" : 9200, "scheme" : "HTTP" }
    ]
  }
}
```

If you are using Elasticsearch service by Elastic, you can just use the `Cloud ID` which is available in the Cloud Console and paste it:

```
{
  "name" : "test",
  "elasticsearch" : {
    "nodes" : [
      { "cloud_id" :
↪"fscrawler:ZXVyb3BlLXdlc3QxLmdjcC5jbG91ZC5lcy5pbyQxZDFlYTk5Njg4Nzc0NWE2YTJiN2NiNzkzMTUzNDhhMyQyOTk
↪" }
    ]
  }
}
```

This ID will be used to automatically generate the right host, port and scheme.

Hint: In the context of Elasticsearch service by Elastic, you will most likely need to provide as well the username and the password. See *Using Credentials (X-Pack)*.

You can define multiple nodes:

```
{
  "name" : "test",
  "elasticsearch" : {
    "nodes" : [
      { "host" : "mynode1.mycompany.com", "port" : 9200, "scheme" : "HTTP" },
      { "host" : "mynode2.mycompany.com", "port" : 9200, "scheme" : "HTTP" },
      { "host" : "mynode3.mycompany.com", "port" : 9200, "scheme" : "HTTP" }
    ]
  }
}
```

Note: New in version 2.2: you can use HTTPS instead of default HTTP.

```
{
  "name" : "test",
  "elasticsearch" : {
    "nodes" : [
      { "host" : "CLUSTERID.eu-west-1.aws.found.io", "port" : 9243, "scheme" : "HTTPS
↪" }
    ]
  }
}
```

For more information, read *SSL Configuration*.

16.5 Using Credentials (X-Pack)

New in version 2.2.

If you secured your elasticsearch cluster with X-Pack, you can provide `username` and `password` to FSCrawler:

```
{
  "name" : "test",
  "elasticsearch" : {
```

(continues on next page)

(continued from previous page)

```
    "username" : "elastic",
    "password" : "changeme"
  }
}
```

Warning: For the current version, the elasticsearch password is stored in plain text in your job setting file.

A better practice is to only set the username or pass it with `--username elastic` option when starting FSCrawler.

If the password is not defined, you will be prompted when starting the job:

```
22:46:42,528 INFO  [f.p.e.c.f.FsCrawler] Password for elastic:
```

16.6 SSL Configuration

In order to ingest documents to Elasticsearch over HTTPS based connection, you need to perform additional configuration steps:

Important: Prerequisite: you need to have root CA chain certificate or Elasticsearch server certificate in DER format. DER format files have a `.cer` extension.

1. Logon to server (or client machine) where FSCrawler is running
2. Run:

```
keytool -import -alias <alias name> -keystore " <JAVA_HOME>\lib\security\cacerts" -
↪file <Path of Elasticsearch Server certificate or Root certificate>
```

It will prompt you for the password. Enter the certificate password like `changeit`.

3. Make changes to FSCrawler `_settings.json` file to connect to your Elasticsearch server over HTTPS:

```
{
  "name" : "test",
  "elasticsearch" : {
    "nodes" : [
      {"host" : "localhost", "port" : 9243, "scheme" : "HTTPS" }
    ]
  }
}
```

Tip: If you can not find `keytool`, it probably means that you did not add your `JAVA_HOME/bin` directory to your path.

16.7 Generated fields

FSCrawler may create the following fields depending on configuration and available data:

For more information about meta data, please read the TikaCoreProperties.

Here is a typical JSON document generated by the crawler:

```
{
   "content":"This is a sample text available in page 1\n\nThis second part of the␣
↪text is in Page 2\n\n",
   "meta":{
      "author":"David Pilato",
      "title":"Test Tika title",
      "date":"2016-07-07T16:37:00.000+0000",
      "keywords":[
         "keyword1",
         "  keyword2"
      ],
      "language":"en",
      "description":"Comments",
      "created":"2016-07-07T16:37:00.000+0000"
   },
   "file":{
      "extension":"odt",
      "content_type":"application/vnd.oasis.opendocument.text",
      "created":"2018-07-30T11:35:08.000+0000",
      "last_modified":"2018-07-30T11:35:08.000+0000",
      "last_accessed":"2018-07-30T11:35:08.000+0000",
      "indexing_date":"2018-07-30T11:35:19.781+0000",
      "filesize":6236,
      "filename":"test.odt",
      "url":"file:///tmp/test.odt"
   },
   "path":{
      "root":"7537e4fb47e553f110a1ec312c2537c0",
      "virtual":"/test.odt",
      "real":"/tmp/test.odt"
   }
}
```

16.8 Search examples

You can use the content field to perform full-text search on

```
GET docs/_search
{
  "query" : {
    "match" : {
        "content" : "the quick brown fox"
    }
  }
}
```

You can use meta fields to perform search on.

```
GET docs/_search
{
  "query" : {
    "term" : {
```

(continues on next page)

(continued from previous page)

```
        "file.filename" : "mydocument.pdf"
    }
  }
}
```

Or run some aggregations on top of them, like:

```
GET docs/_search
{
  "size": 0,
  "aggs": {
    "by_extension": {
      "terms": {
        "field": "file.extension"
      }
    }
  }
}
```

CHAPTER 17

REST service

New in version 2.2.

FSCrawler can expose a REST service running at http://127.0.0.1:8080/fscrawler. To activate it, launch FSCrawler with `--rest` option.

17.1 FSCrawler status

To get an overview of the running service, you can call `GET /` endpoint:

```
curl http://127.0.0.1:8080/fscrawler/
```

It will give you a response similar to:

```
{
  "ok" : true,
  "version" : "2.2",
  "elasticsearch" : "5.1.1",
  "settings" : {
    "name" : "fscrawler-rest-tests",
    "fs" : {
      "url" : "/tmp/es",
      "update_rate" : "15m",
      "json_support" : false,
      "filename_as_id" : false,
      "add_filesize" : true,
      "remove_deleted" : true,
      "store_source" : false,
      "index_content" : true,
      "attributes_support" : false,
      "raw_metadata" : true,
      "xml_support" : false,
      "index_folders" : true,
```

(continues on next page)

(continued from previous page)

```
      "lang_detect" : false
    },
    "elasticsearch" : {
      "nodes" : [ {
        "host" : "127.0.0.1",
        "port" : 9200,
        "scheme" : "HTTP"
      } ],
      "index" : "fscrawler-rest-tests_doc",
      "index_folder" : "fscrawler-rest-tests_folder",
      "bulk_size" : 100,
      "flush_interval" : "5s",
      "byte_size" : "10mb",
      "username" : "elastic"
    },
    "rest" : {
      "scheme" : "HTTP",
      "host" : "127.0.0.1",
      "port" : 8080,
      "endpoint" : "fscrawler"
    }
  }
}
```

17.2 Uploading a binary document

To upload a binary, you can call `POST /_upload` endpoint:

```
echo "This is my text" > test.txt
curl -F "file=@test.txt" "http://127.0.0.1:8080/fscrawler/_upload"
```

It will give you a response similar to:

```
{
  "ok" : true,
  "filename" : "test.txt",
  "url" : "http://127.0.0.1:9200/fscrawler-rest-tests_doc/doc/
↪dd18bf3a8ea2a3e53e2661c7fb53534"
}
```

The `url` represents the elasticsearch address of the indexed document. If you call:

```
curl http://127.0.0.1:9200/fscrawler-rest-tests_doc/doc/
↪dd18bf3a8ea2a3e53e2661c7fb53534?pretty
```

You will get back your document as it has been stored by elasticsearch:

```
{
  "_index" : "fscrawler-rest-tests_doc",
  "_type" : "_doc",
  "_id" : "dd18bf3a8ea2a3e53e2661c7fb53534",
  "_version" : 1,
  "found" : true,
  "_source" : {
```

(continues on next page)

(continued from previous page)

```
    "content" : "This file contains some words.\n",
    "meta" : {
      "raw" : {
        "X-Parsed-By" : "org.apache.tika.parser.DefaultParser",
        "Content-Encoding" : "ISO-8859-1",
        "Content-Type" : "text/plain; charset=ISO-8859-1"
      }
    },
    "file" : {
      "extension" : "txt",
      "content_type" : "text/plain; charset=ISO-8859-1",
      "indexing_date" : "2017-01-04T21:01:08.043",
      "filename" : "test.txt"
    },
    "path" : {
      "virtual" : "test.txt",
      "real" : "test.txt"
    }
  }
}
```

If you started FSCrawler in debug mode with `--debug` or if you pass `debug=true` query parameter, then the response will be much more complete:

```
echo "This is my text" > test.txt
curl -F "file=@test.txt" "http://127.0.0.1:8080/fscrawler/_upload?debug=true"
```

will give

```
{
  "ok" : true,
  "filename" : "test.txt",
  "url" : "http://127.0.0.1:9200/fscrawler-rest-tests_doc/doc/
↪dd18bf3a8ea2a3e53e2661c7fb53534",
  "doc" : {
    "content" : "This file contains some words.\n",
    "meta" : {
      "raw" : {
        "X-Parsed-By" : "org.apache.tika.parser.DefaultParser",
        "Content-Encoding" : "ISO-8859-1",
        "Content-Type" : "text/plain; charset=ISO-8859-1"
      }
    },
    "file" : {
      "extension" : "txt",
      "content_type" : "text/plain; charset=ISO-8859-1",
      "indexing_date" : "2017-01-04T14:05:10.325",
      "filename" : "test.txt"
    },
    "path" : {
      "virtual" : "test.txt",
      "real" : "test.txt"
    }
  }
}
```

17.3 Simulate Upload

If you want to get back the extracted content and its metadata but without indexing into elasticsearch you can use `simulate=true` query parameter:

```
echo "This is my text" > test.txt
curl -F "file=@test.txt" "http://127.0.0.1:8080/fscrawler/_upload?debug=true&
↪simulate=true"
```

17.4 Document ID

By default, FSCrawler encodes the filename to generate an id. Which means that if you send 2 files with the same filename `test.txt`, the second one will overwrite the first one because they will both share the same ID.

You can force any id you wish by adding `id=YOUR_ID` in the form data:

```
echo "This is my text" > test.txt
curl -F "file=@test.txt" -F "id=my-test" "http://127.0.0.1:8080/fscrawler/_upload"
```

There is a specific id named `_auto_` where the ID will be autogenerated by elasticsearch. It means that sending twice the same file will result in 2 different documents indexed.

17.5 Additional tags

Add custom tags to the document. In case you want to do filtering on those tags (examples are `projectId` or `tenantId`). These tags can be assigned to an `external` object field. As you can see in the json, you are able to overwrite the `content` field. `meta`, `file` and `path` fields can be overwritten as well. To upload a binary with additional tags, you can call `POST /_upload` endpoint:

```
{
  "content": "OVERWRITE CONTENT",
  "external": {
    "tenantId": 23,
    "projectId": 34,
    "description": "these are additional tags"
  }
}
```

```
echo "This is my text" > test.txt
echo "{\"content\":\"OVERWRITE CONTENT\",\"external\"{\"tenantId\": 23,\"projectId\":␣
↪34,\"description\":\"these are additional tags\"}}" > tags.txt
curl -F "file=@test.txt" -F "tags=@tags.txt" "http://127.0.0.1:8080/fscrawler/_upload"
```

The field `external` doesn't necessarily be a flat structure. This is a more advanced example:

```
{
  "external": {
    "tenantId" : 23,
    "company": "shoe company",
    "projectId": 34,
    "project": "business development",
    "daysOpen": [
```

(continues on next page)

(continued from previous page)

```
      "Mon",
      "Tue",
      "Wed",
      "Thu",
      "Fri"
    ],
    "products": [
      {
        "brand": "nike",
        "size": 41,
        "sub": "Air MAX"
      },
      {
        "brand": "reebok",
        "size": 43,
        "sub": "Pump"
      }
    ]
  }
}
```

Attention: Only standard *FSCrawler fields* can be set outside `external` field name.

17.6 REST settings

Here is a list of REST service settings (under `rest.` prefix)‘:

Name	Default value	Documentation
`rest.scheme`	`http`	Scheme. Can be either `http` or `https`
`rest.host`	`127.0.0.1`	Bound host
`rest.port`	`8080`	Bound port
`rest.endpoint`	`fscrawler`	Endpoint

Tip: Most *Local FS settings* (under `fs.*` in the settings file) also affect the REST service, e.g. `fs.indexed_chars`. Local FS settings that do **not** affect the REST service are those such as `url`, `update_rate`, `includes`, `excludes`.

REST service is running at http://127.0.0.1:8080/fscrawler by default.

You can change it using `rest` settings:

```
{
  "name" : "test",
  "rest" : {
    "scheme" : "HTTP",
    "host" : "192.168.0.1",
    "port" : 8180,
    "endpoint" : "my_fscrawler"
  }
}
```

It also means that if you are running more than one instance of FS crawler locally, you can (must) change the `port`.

CHAPTER 18

Building the project

This project is built with Maven.

18.1 Build the artifact

To build the project, run:

```
mvn clean package
```

The final artifact is available in `distribution/target` directory.

Tip: To build it faster (without tests), run:

```
mvn clean package -DskipTests
```

18.2 Run tests with an external cluster

To run the test suite against an elasticsearch instance running locally, just run:

```
mvn verify
```

Tip: If you don't want to rebuild everything (ie. you just touch test classes), run:

```
mvn -pl fr.pilato.elasticsearch.crawler:fscrawler-it verify
```

If elasticsearch is not running yet on `http://localhost:9200`, FSCrawler project will run a Docker instance before the tests start.

Hint: If you are using a secured instance, use `tests.cluster.user`, `tests.cluster.pass` and `tests.cluster.scheme`:

```
mvn verify \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.cluster.scheme=HTTPS \
```

Hint: To run tests against another instance (ie. running on Elasticsearch service by Elastic, you can also use `tests.cluster.host` and `tests.cluster.port` to set where elasticsearch is running:

```
mvn verify \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.cluster.scheme=HTTPS \
    -Dtests.cluster.host=XYZ.es.io:9243 \
    -Dtests.cluster.port=9243
```

Or even easier, you can use the `Cloud ID` available on you Cloud Console:

```
mvn verify \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.cluster.cloud_
↪id=fscrawler:ZXVyb3BlLXdlc3QxLmdjcC5jbG91ZC5lcy5pbyQxZDFlYTk5Njg4Nzc0NWE2YTJiN2NiNzkzMTUzNDhhMyQyO
```

18.3 Check for vulnerabilities (CVE)

The project is using OSS Sonatype service to check for known vulnerabilities. This is ran during the `verify` phase.

Sonatype provides this service but with a anonymous account, you might be limited by the number of tests you can run during a given period.

If you have an existing account, you can use it to bypass this limit for anonymous users by setting `sonatype.username` and `sonatype.password`:

```
mvn verify -DskipTests \
    -Dsonatype.username=youremail@domain.com \
    -Dsonatype.password=yourverysecuredpassword
```

CHAPTER 19

Writing documentation

This project uses ReadTheDocs to build and serve the documentation.

If you want to run the generation of documentation (recommended!), you need to have Python installed. Then install `sphinx` $ pip install sphinx sphinx-autobuild

Assuming you have Python already, install Sphinx:

```
$ pip install sphinx sphinx-autobuild
```

Go to the `docs` directory and build the html documentation:

```
$ cd docs
$ make html
```

Just open then `target/html/index.html` page in your browser.

Hint: You can hot reload your changes by using `sphinx-autobuild`:

```
$ sphinx-autobuild source target/html
```

Then just edit the documentation and look for your changes at http://127.0.0.1:8000

To learn more about the reStructuredText format, please look at the basic guide.

CHAPTER 20

Release the project

To release the project, run:

```
$ release.sh
```

And follow the instructions.

Note: Only developers with write rights to the sonatype repository under `fr.pilato` space can perform the release.

CHAPTER 21

License

Important: This software is licensed under the Apache 2 license, quoted below.

CHAPTER 22

Incompatible 3rd party library licenses

Some libraries are not Apache2 compatible. Therefore they are not packaged with FSCrawler so you need to download and add manually them to the `lib` directory:

- for JBIG2 images, you need to add levigo-jbig2-imageio:2.0 library
- for TIFF images, you need to add jai-imageio-core:1.4.0 library
- for JPEG 2000 (JPX) images, you need to add jai-imageio-jpeg2000:1.3.0 library

See pdfbox documentation for more details.