FSCrawler Documentation

Release 2.10-SNAPSHOT

David Pilato

Apr 25, 2024

INSTALLATION GUIDE

Warning: This documentation is for the version of FSCrawler currently under development. Were you looking for the documentation of the latest stable version?

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/FTP crawling.
- REST interface to let you "upload" your binary documents to elasticsearch.

Note: FS Crawler 2.10-SNAPSHOT is using Tika 2.9.2 and is tested against:

- Elasticsearch 8.13.2.
- Elasticsearch 7.17.19.
- Elasticsearch 6.8.23. (Deprecated)

CHAPTER ONE

DOWNLOAD FSCRAWLER

Depending on your Elasticsearch cluster version, you can download FSCrawler 2.10 using the following links from Sonatype.

The filename ends with `.zip`.

Warning: This is a **SNAPSHOT** version. You can also download a **stable** version from Maven Central.

See *Directory layout* to know more about the content of the distribution.

CHAPTER

TWO

USING DOCKER

Pull the Docker image:

```
docker pull dadoonet/fscrawler
```

Note: This image is very big (1.2+gb) as it contains Tesseract and all the trained language data. If you don't want to use OCR at all, you can use a smaller image (around 530mb) by pulling instead `dadoonet/fscrawler:noocr`

```
docker pull dadoonet/fscrawler:noocr
```

Let say your documents are located in `~/tmp` dir and you want to store your fscrawler jobs in `~/.fscrawler`. You can run FSCrawler with:

```
docker run -it --rm \
     -v ~/.fscrawler:/root/.fscrawler \
     -v ~/tmp:/tmp/es:ro \
     dadoonet/fscrawler fscrawler job_name
```

On the first run, if the job does not exist yet in `~/.fscrawler`, FSCrawler will ask you if you want to create it:

```
10:16:53,880 INFO  [f.p.e.c.f.c.BootstrapChecks] Memory [Free/Total=Percent]: HEAP [67.
↪3mb/876.5mb=7.69%], RAM [2.1gb/3.8gb=55.43%], Swap [1023.9mb/1023.9mb=100.0%].
10:16:53,899 WARN  [f.p.e.c.f.c.FsCrawlerCli] job [job_name] does not exist
10:16:53,900 INFO  [f.p.e.c.f.c.FsCrawlerCli] Do you want to create it (Y/N)?
y
10:16:56,745 INFO  [f.p.e.c.f.c.FsCrawlerCli] Settings have been created in [/root/.
↪fscrawler/job_name/_settings.yaml]. Please review and edit before relaunch
```

Note: The configuration file is actually stored on your machine in `~/.fscrawler/job_name/_settings.yaml`. Remember to change the URL of your elasticsearch instance as the container won't be able to see it running under the default `127.0.0.1`. You will need to use the actual IP address of the host.

If you need to add a 3rd party library (jar) or your Tika custom jar, you can put it in a `external` directory and mount it as well:

```
docker run -it --rm \
     -v ~/.fscrawler:/root/.fscrawler \
     -v ~/tmp:/tmp/es:ro \
```

(continues on next page)

(continued from previous page)

```
    -v "$PWD/external:/usr/share/fscrawler/external" \
    dadoonet/fscrawler fscrawler job_name
```

CHAPTER

THREE

USING DOCKER COMPOSE

In this section, the following directory layout is assumed:

```
.
├── config
│   └── job_name
│       └── _settings.yaml
├── data
│   └── <your files>
├── external
│   └── <3rd party jars if needed>
├── logs
│   └── <fscrawler logs>
└── docker-compose.yml
```

3.1 With Elasticsearch

Here is a typical _settings.yaml, you can use to connect FSCrawler with Elasticsearch when running with docker compose:

```
---
name: "idx"
fs:
  indexed_chars: 100%
  lang_detect: true
  continue_on_error: true
  ocr:
    language: "eng"
    enabled: true
    pdf_strategy: "ocr_and_text"
elasticsearch:
  nodes:
    - url: "https://elasticsearch:9200"
  username: "elastic"
  password: "changeme"
  ssl_verification: false
rest :
  url: "http://fscrawler:8080"
```

Note: The configuration shown above is also meant to start the REST interface. It also activates the full indexation of documents, lang detection and ocr using english. You can adapt this example for your needs.

And, prepare the following `docker-compose.yml`. You will find this example in the `contrib/docker-compose-example-elasticsearch` project directory.

```yaml
---
version: "2.2"

services:
  setup:
    image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
    volumes:
      - certs:/usr/share/elasticsearch/config/certs
    user: "0"
    command: >
      bash -c '
        if [ x${ELASTIC_PASSWORD} == x ]; then
          echo "Set the ELASTIC_PASSWORD environment variable in the .env file";
          exit 1;
        elif [ x${KIBANA_PASSWORD} == x ]; then
          echo "Set the KIBANA_PASSWORD environment variable in the .env file";
          exit 1;
        fi;
        if [ ! -f certs/ca.zip ]; then
          echo "Creating CA";
          bin/elasticsearch-certutil ca --silent --pem -out config/certs/ca.zip;
          unzip config/certs/ca.zip -d config/certs;
        fi;
        if [ ! -f certs/certs.zip ]; then
          echo "Creating certs";
          echo -ne \
          "instances:\n"\
          "  - name: elasticsearch\n"\
          "    dns:\n"\
          "      - elasticsearch\n"\
          "      - localhost\n"\
          "    ip:\n"\
          "      - 127.0.0.1\n"\
          > config/certs/instances.yml;
          bin/elasticsearch-certutil cert --silent --pem -out config/certs/certs.zip --
↪in config/certs/instances.yml --ca-cert config/certs/ca/ca.crt --ca-key config/certs/
↪ca/ca.key;
          unzip config/certs/certs.zip -d config/certs;
        fi;
        echo "Setting file permissions"
        chown -R root:root config/certs;
        find . -type d -exec chmod 750 \{\} \;;
        find . -type f -exec chmod 640 \{\} \;;
        echo "Waiting for Elasticsearch availability";
        until curl -s --cacert config/certs/ca/ca.crt https://elasticsearch:9200 | grep -
↪q "missing authentication credentials"; do sleep 30; done;
```

(continues on next page)

(continued from previous page)

```
        echo "Setting kibana_system password";
        until curl -s -X POST --cacert config/certs/ca/ca.crt -u elastic:${ELASTIC_
↪PASSWORD} -H "Content-Type: application/json" https://elasticsearch:9200/_security/
↪user/kibana_system/_password -d "{\"password\":\"${KIBANA_PASSWORD}\"}" | grep -q "^{}
↪"; do sleep 10; done;
        echo "All done!";
      '
    healthcheck:
      test: ["CMD-SHELL", "[ -f config/certs/elasticsearch/elasticsearch.crt ]"]
      interval: 1s
      timeout: 5s
      retries: 120

  elasticsearch:
    depends_on:
      setup:
        condition: service_healthy
    image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
    volumes:
      - certs:/usr/share/elasticsearch/config/certs
      - esdata:/usr/share/elasticsearch/data
    ports:
      - ${ES_PORT}:9200
    environment:
      - node.name=elasticsearch
      - cluster.name=${CLUSTER_NAME}
      - cluster.initial_master_nodes=elasticsearch
      - ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
      - bootstrap.memory_lock=true
      - xpack.security.enabled=true
      - xpack.security.http.ssl.enabled=true
      - xpack.security.http.ssl.key=certs/elasticsearch/elasticsearch.key
      - xpack.security.http.ssl.certificate=certs/elasticsearch/elasticsearch.crt
      - xpack.security.http.ssl.certificate_authorities=certs/ca/ca.crt
      - xpack.security.http.ssl.verification_mode=certificate
      - xpack.security.transport.ssl.enabled=true
      - xpack.security.transport.ssl.key=certs/elasticsearch/elasticsearch.key
      - xpack.security.transport.ssl.certificate=certs/elasticsearch/elasticsearch.crt
      - xpack.security.transport.ssl.certificate_authorities=certs/ca/ca.crt
      - xpack.security.transport.ssl.verification_mode=certificate
      - xpack.license.self_generated.type=${LICENSE}
    mem_limit: ${MEM_LIMIT}
    ulimits:
      memlock:
        soft: -1
        hard: -1
    healthcheck:
      test:
        [
          "CMD-SHELL",
          "curl -s --cacert config/certs/ca/ca.crt https://localhost:9200 | grep -q
↪'missing authentication credentials'",
```

(continues on next page)

(continued from previous page)

```
        ]
      interval: 10s
      timeout: 10s
      retries: 120

  kibana:
    depends_on:
      elasticsearch:
        condition: service_healthy
    image: docker.elastic.co/kibana/kibana:${STACK_VERSION}
    volumes:
      - certs:/usr/share/kibana/config/certs
      - kibanadata:/usr/share/kibana/data
    ports:
      - ${KIBANA_PORT}:5601
    environment:
      - SERVERNAME=kibana
      - ELASTICSEARCH_HOSTS=https://elasticsearch:9200
      - ELASTICSEARCH_USERNAME=kibana_system
      - ELASTICSEARCH_PASSWORD=${KIBANA_PASSWORD}
      - ELASTICSEARCH_SSL_CERTIFICATEAUTHORITIES=config/certs/ca/ca.crt
      - ENTERPRISESEARCH_HOST=http://enterprisesearch:${ENTERPRISE_SEARCH_PORT}
    mem_limit: ${MEM_LIMIT}
    healthcheck:
      test:
        [
          "CMD-SHELL",
          "curl -s -I http://localhost:5601 | grep -q 'HTTP/1.1 302 Found'",
        ]
      interval: 10s
      timeout: 10s
      retries: 120

  # FSCrawler
  fscrawler:
    image: dadoonet/fscrawler:$FSCRAWLER_VERSION
    container_name: fscrawler
    restart: always
    volumes:
      - ../../test-documents/src/main/resources/documents/:/tmp/es:ro
      - ${PWD}/config:/root/.fscrawler
      - ${PWD}/logs:/usr/share/fscrawler/logs
      - ${PWD}/external:/usr/share/fscrawler/external
    depends_on:
      elasticsearch:
        condition: service_healthy
    ports:
      - ${FSCRAWLER_PORT}:8080
    command: fscrawler idx --restart --rest

volumes:
  certs:
```

(continues on next page)

(continued from previous page)

```
    driver: local
  esdata:
    driver: local
  kibanadata:
    driver: local
```

Note: The configuration shown above is also meant to start Kibana. You can skip that part if you don't need it.

Then, you can run the full stack, including FSCrawler.

```
docker-compose up -d
```

3.2 With Enterprise Search (Workplace Search)

Here is a typical `_settings.yaml`, you can use to connect FSCrawler with Workplace Search when running with docker compose:

```
---
name: "idx"
fs:
  indexed_chars: 100%
  lang_detect: true
  continue_on_error: true
  ocr:
    language: "eng"
    enabled: true
    pdf_strategy: "ocr_and_text"
elasticsearch:
  nodes:
    - url: "https://elasticsearch:9200"
  username: "elastic"
  password: "changeme"
  ssl_verification: false
workplace_search:
  server: "http://enterprisesearch:3002"
```

And, prepare the following `docker-compose.yml`. You will find this example in the `contrib/docker-compose-example-workplace` project directory.

```
---
version: "2.2"

services:
  setup:
    image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
    volumes:
      - certs:/usr/share/elasticsearch/config/certs
    user: "0"
    command: >
```

(continues on next page)

(continued from previous page)

```
      bash -c '
        if [ x${ELASTIC_PASSWORD} == x ]; then
          echo "Set the ELASTIC_PASSWORD environment variable in the .env file";
          exit 1;
        elif [ x${KIBANA_PASSWORD} == x ]; then
          echo "Set the KIBANA_PASSWORD environment variable in the .env file";
          exit 1;
        fi;
        if [ ! -f certs/ca.zip ]; then
          echo "Creating CA";
          bin/elasticsearch-certutil ca --silent --pem -out config/certs/ca.zip;
          unzip config/certs/ca.zip -d config/certs;
        fi;
        if [ ! -f certs/certs.zip ]; then
          echo "Creating certs";
          echo -ne \
          "instances:\n"\
          "  - name: elasticsearch\n"\
          "    dns:\n"\
          "      - elasticsearch\n"\
          "      - localhost\n"\
          "    ip:\n"\
          "      - 127.0.0.1\n"\
          > config/certs/instances.yml;
          bin/elasticsearch-certutil cert --silent --pem -out config/certs/certs.zip --
↪in config/certs/instances.yml --ca-cert config/certs/ca/ca.crt --ca-key config/certs/
↪ca/ca.key;
          unzip config/certs/certs.zip -d config/certs;
        fi;
        echo "Setting file permissions"
        chown -R root:root config/certs;
        find . -type d -exec chmod 750 \{\} \;;
        find . -type f -exec chmod 640 \{\} \;;
        echo "Waiting for Elasticsearch availability";
        until curl -s --cacert config/certs/ca/ca.crt https://elasticsearch:9200 | grep -
↪q "missing authentication credentials"; do sleep 30; done;
        echo "Setting kibana_system password";
        until curl -s -X POST --cacert config/certs/ca/ca.crt -u elastic:${ELASTIC_
↪PASSWORD} -H "Content-Type: application/json" https://elasticsearch:9200/_security/
↪user/kibana_system/_password -d "{\"password\":\"${KIBANA_PASSWORD}\"}" | grep -q "^{}
↪"; do sleep 10; done;
        echo "All done!";
      '
    healthcheck:
      test: ["CMD-SHELL", "[ -f config/certs/elasticsearch/elasticsearch.crt ]"]
      interval: 1s
      timeout: 5s
      retries: 120

  elasticsearch:
    depends_on:
      setup:
```

(continues on next page)

(continued from previous page)

```
        condition: service_healthy
    image: docker.elastic.co/elasticsearch/elasticsearch:${STACK_VERSION}
    volumes:
      - certs:/usr/share/elasticsearch/config/certs
      - esdata:/usr/share/elasticsearch/data
    ports:
      - ${ES_PORT}:9200
    environment:
      - node.name=elasticsearch
      - cluster.name=${CLUSTER_NAME}
      - cluster.initial_master_nodes=elasticsearch
      - ELASTIC_PASSWORD=${ELASTIC_PASSWORD}
      - bootstrap.memory_lock=true
      - xpack.security.enabled=true
      - xpack.security.http.ssl.enabled=true
      - xpack.security.http.ssl.key=certs/elasticsearch/elasticsearch.key
      - xpack.security.http.ssl.certificate=certs/elasticsearch/elasticsearch.crt
      - xpack.security.http.ssl.certificate_authorities=certs/ca/ca.crt
      - xpack.security.http.ssl.verification_mode=certificate
      - xpack.security.transport.ssl.enabled=true
      - xpack.security.transport.ssl.key=certs/elasticsearch/elasticsearch.key
      - xpack.security.transport.ssl.certificate=certs/elasticsearch/elasticsearch.crt
      - xpack.security.transport.ssl.certificate_authorities=certs/ca/ca.crt
      - xpack.security.transport.ssl.verification_mode=certificate
      - xpack.license.self_generated.type=${LICENSE}
    mem_limit: ${MEM_LIMIT}
    ulimits:
      memlock:
        soft: -1
        hard: -1
    healthcheck:
      test:
        [
          "CMD-SHELL",
          "curl -s --cacert config/certs/ca/ca.crt https://localhost:9200 | grep -q
↪'missing authentication credentials'",
        ]
      interval: 10s
      timeout: 10s
      retries: 120

  kibana:
    depends_on:
      elasticsearch:
        condition: service_healthy
    image: docker.elastic.co/kibana/kibana:${STACK_VERSION}
    volumes:
      - certs:/usr/share/kibana/config/certs
      - kibanadata:/usr/share/kibana/data
    ports:
      - ${KIBANA_PORT}:5601
    environment:
```

(continues on next page)

(continued from previous page)

```
      - SERVERNAME=kibana
      - ELASTICSEARCH_HOSTS=https://elasticsearch:9200
      - ELASTICSEARCH_USERNAME=kibana_system
      - ELASTICSEARCH_PASSWORD=${KIBANA_PASSWORD}
      - ELASTICSEARCH_SSL_CERTIFICATEAUTHORITIES=config/certs/ca/ca.crt
      - ENTERPRISESEARCH_HOST=http://enterprisesearch:${ENTERPRISE_SEARCH_PORT}
    mem_limit: ${MEM_LIMIT}
    healthcheck:
      test:
        [
          "CMD-SHELL",
          "curl -s -I http://localhost:5601 | grep -q 'HTTP/1.1 302 Found'",
        ]
      interval: 10s
      timeout: 10s
      retries: 120

  enterprisesearch:
    depends_on:
      elasticsearch:
        condition: service_healthy
      kibana:
        condition: service_healthy
    image: docker.elastic.co/enterprise-search/enterprise-search:${STACK_VERSION}
    volumes:
      - certs:/usr/share/enterprise-search/config/certs
      - enterprisesearchdata:/usr/share/enterprise-search/config
    ports:
      - ${ENTERPRISE_SEARCH_PORT}:3002
    environment:
      - SERVERNAME=enterprisesearch
      - secret_management.encryption_keys=[${ENCRYPTION_KEYS}]
      - allow_es_settings_modification=true
      - elasticsearch.host=https://elasticsearch:9200
      - elasticsearch.username=elastic
      - elasticsearch.password=${ELASTIC_PASSWORD}
      - elasticsearch.ssl.enabled=true
      - elasticsearch.ssl.certificate_authority=/usr/share/enterprise-search/config/
↪certs/ca/ca.crt
      - kibana.external_url=http://kibana:5601
    mem_limit: ${MEM_LIMIT}
    healthcheck:
      test:
        [
          "CMD-SHELL",
          "curl -s -I http://localhost:3002 | grep -q 'HTTP/1.1 302 Found'",
        ]
      interval: 10s
      timeout: 10s
      retries: 120

  # Apache Httpd service (to serve local files)
```

(continues on next page)

(continued from previous page)

```
  httpd:
    image: httpd:2.4
    restart: on-failure
    volumes:
      - ../../test-documents/src/main/resources/documents/:/usr/local/apache2/htdocs/:ro
    ports:
      - 80:80
    healthcheck:
      test:
        [
          "CMD-SHELL",
          "curl -s -I http://localhost | grep -q 'HTTP/1.1 302 Found'",
        ]
      interval: 10s
      timeout: 10s
      retries: 120

  # FSCrawler
  fscrawler:
    image: dadoonet/fscrawler:$FSCRAWLER_VERSION
    container_name: fscrawler
    restart: on-failure
    volumes:
      - ../../test-documents/src/main/resources/documents/:/tmp/es:ro
      - ${PWD}/config:/root/.fscrawler
      - ${PWD}/logs:/usr/share/fscrawler/logs
      - ${PWD}/external:/usr/share/fscrawler/external
    depends_on:
      enterprisesearch:
        condition: service_healthy
    command: fscrawler idx --restart

volumes:
  certs:
    driver: local
  enterprisesearchdata:
    driver: local
  esdata:
    driver: local
  kibanadata:
    driver: local
```

Note: The configuration shown above is also meant to start a local HTTP server which will serve your local files when you click on a result from the Workplace Search interface.

Then, you can run the full stack, including FSCrawler and the HTTP Web Server.

```
docker-compose up -d
```

FSCrawler will index all the documents and then exit.

When the FSCrawler container has stopped, you can just open the search interface and start to search for your local documents. You might need to be authenticated first in Kibana. You can also open Kibana to access the Workplace Search configuration and modify the source which has been created by FSCrawler.

CHAPTER FOUR

RUNNING AS A SERVICE ON WINDOWS

Create a `fscrawlerRunner.bat` as:

```
set JAVA_HOME=c:\Program Files\Java\jdk15.0.1
set FS_JAVA_OPTS=-Xmx2g -Xms2g
/Elastic/fscrawler/bin/fscrawler.bat --config_dir /Elastic/fscrawler data >> /Elastic/
↪logs/fscrawler.log 2>&1
```

Then use `fscrawlerRunner.bat` to create your windows service.

CHAPTER

FIVE

GETTING STARTED

You need to have at least **Java 11** and have properly configured JAVA_HOME to point to your Java installation directory. For example on MacOS if you are using sdkman you can define in your ~/.bash_profile file:

```
export JAVA_HOME="~/.sdkman/candidates/java/current"
```

5.1 Start FSCrawler

Start FSCrawler with:

```
bin/fscrawler job_name
```

FSCrawler will read a local file (default to ~/.fscrawler/{job_name}/_settings.yaml). 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.yaml]. 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
```

5.2 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.

5.3 Ignoring folders

If you would like to ignore some folders to be scanned, just add a `.fscrawlerignore` file in it. The folder content and all sub folders will be ignored.

For more information, read *Includes and excludes*.

CHAPTER

SIX

TUTORIAL

This tutorial use case is:

Search for the resumes (PDF or Word file which resides in One drive or local) and search for anything in the content using Kibana. For example location worked or the previous company, etc.

6.1 Prerequisites

- Java 11+ must be installed
- `JAVA_HOME` must be defined

6.2 Install Elastic stack

- Download Elasticsearch
- Download Kibana
- Start Elasticsearch server
- Start Kibana server
- Check that Kibana is running by opening http://localhost:5601

6.3 Start FSCrawler

- Download FSCrawler. See *Download FSCrawler*.
- Open a terminal and navigate to the `fscrawler` folder.
- Type:

```
# On Linux/Mac
bin/fscrawler resumes
# On Windows
.\bin\fscrawler resumes
```

- It will ask “Do you want to create it (Y/N)?”. Answer `Y`.
- Go to the FSCrawler configuration folder to edit the job configuration. The FSCrawler configuration folder named `.fscrawler` is by default in the user home directory, like `C:\Users\myuser` on Windows platform or ~ on Linux/MacOS. In this folder, you will find another folder named `resumes`. Enter this folder:

```
# On Linux/Mac
cd ~/.fscrawler/resumes
# On Windows
cd C:\Users\myuser\.fscrawler\resumes
```

- Edit the `_settings.yaml` file which is in this folder and change the `url` value to your folder which contains the resumes you would like to index:

```
---
name: "resumes"
fs:
  # On Linux
  url: "/path/to/resumes"
  # On Windows
  url: "c:\\path\\to\\resumes"
```

- Start again FSCrawler:

```
# On Linux/Mac
bin/fscrawler resumes
# On Windows
.\bin\fscrawler resumes
```

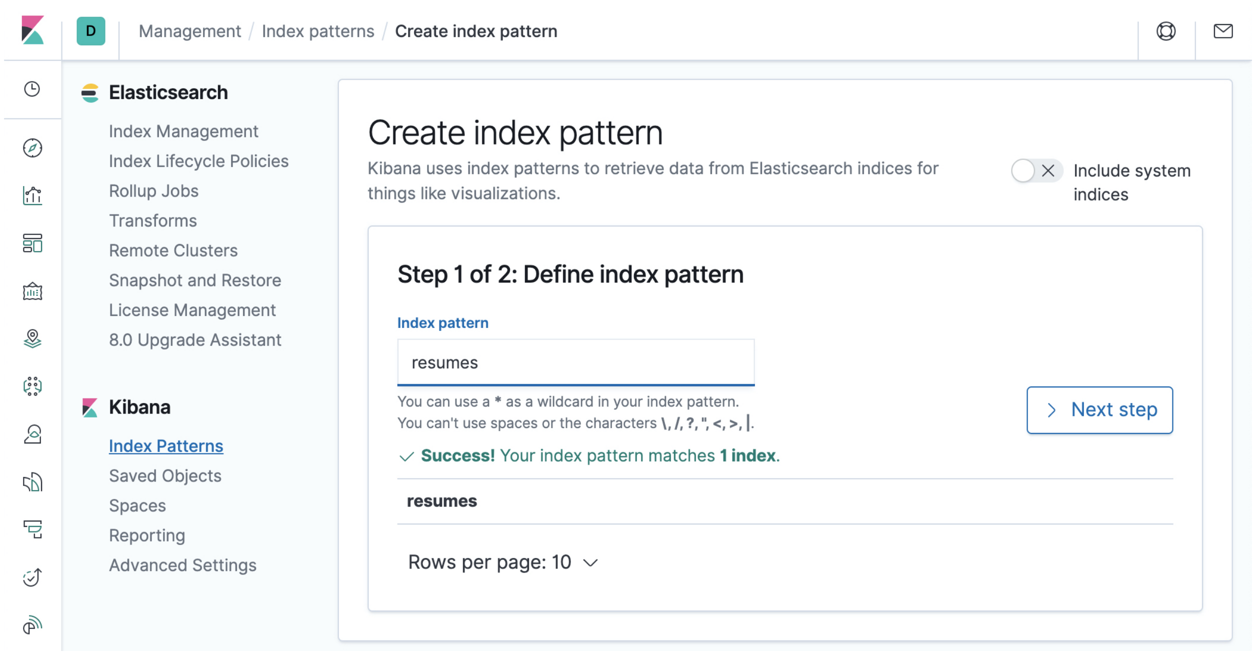
FSCrawler should index all the documents inside your directory.

Note: If you want to start again reindexing from scratch instead of monitoring the changes, stop FSCrawler, restart it with the `--restart` option:

```
# On Linux/Mac
bin/fscrawler resumes --restart
# On Windows
.\bin\fscrawler resumes --restart
```

6.4 Create Index pattern

- Open Kibana
- Go to the Management page
- Open the Index Patterns page under Kibana settings.
- Click on `Create index pattern`
- Type `resumes` in the input box. Don't forget to remove the star * that is automatically added by default by Kibana.


Management / Index patterns / Create index pattern

Elasticsearch
Index Management
Index Lifecycle Policies
Rollup Jobs
Transforms
Remote Clusters
Snapshot and Restore
License Management
8.0 Upgrade Assistant

Kibana
Index Patterns
Saved Objects
Spaces
Reporting
Advanced Settings

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

Include system indices

Step 1 of 2: Define index pattern

Index pattern: resumes

You can use a * as a wildcard in your index pattern.
You can't use spaces or the characters \, /, ?, ", <, >, |.

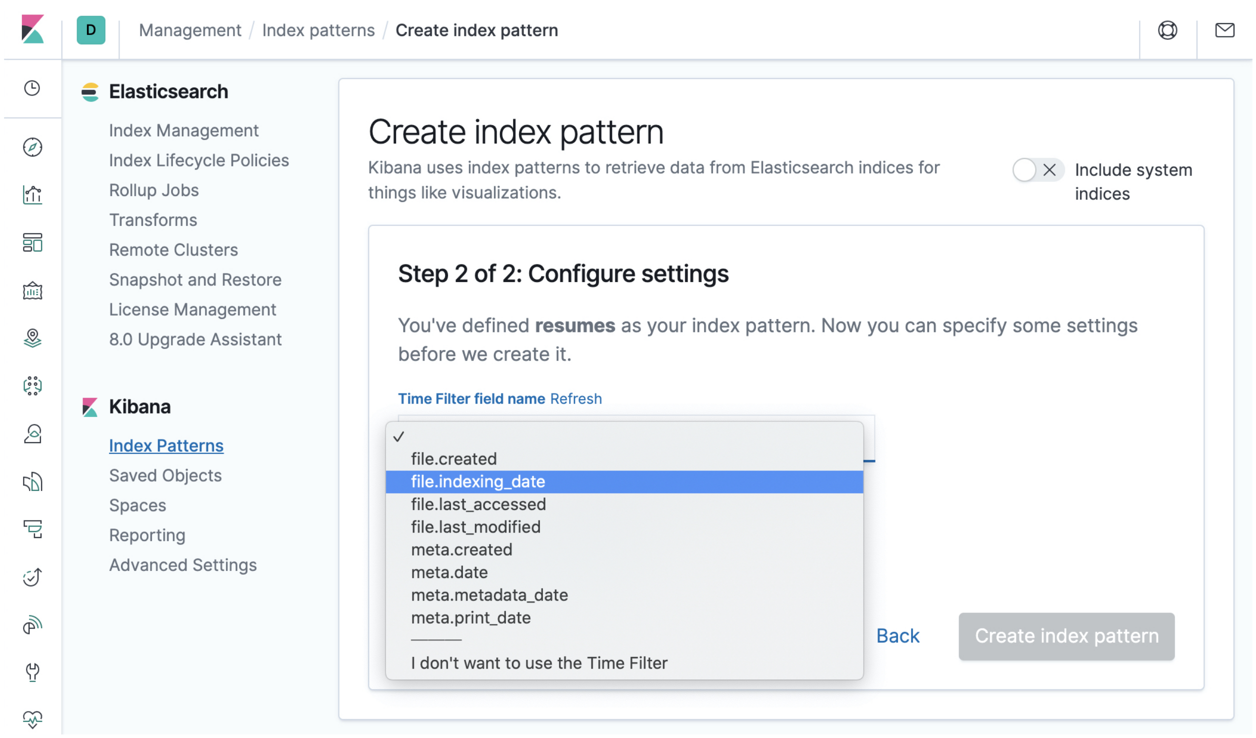
Next step

✓ Success! Your index pattern matches 1 index.

resumes

Rows per page: 10

- Choose the date field you'd like to use if you want to be able to filter documents by date. Use `file.created` field if you want to filter by file creation date, `file.last_modified` to filter by last modification date or `file.indexing_date` if you want to filter by the date when the document has been indexed into elasticsearch. You can also choose not to use the time filter (the last option).


Management / Index patterns / Create index pattern

Elasticsearch
Index Management
Index Lifecycle Policies
Rollup Jobs
Transforms
Remote Clusters
Snapshot and Restore
License Management
8.0 Upgrade Assistant

Kibana
Index Patterns
Saved Objects
Spaces
Reporting
Advanced Settings

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

Include system indices

Step 2 of 2: Configure settings

You've defined **resumes** as your index pattern. Now you can specify some settings before we create it.

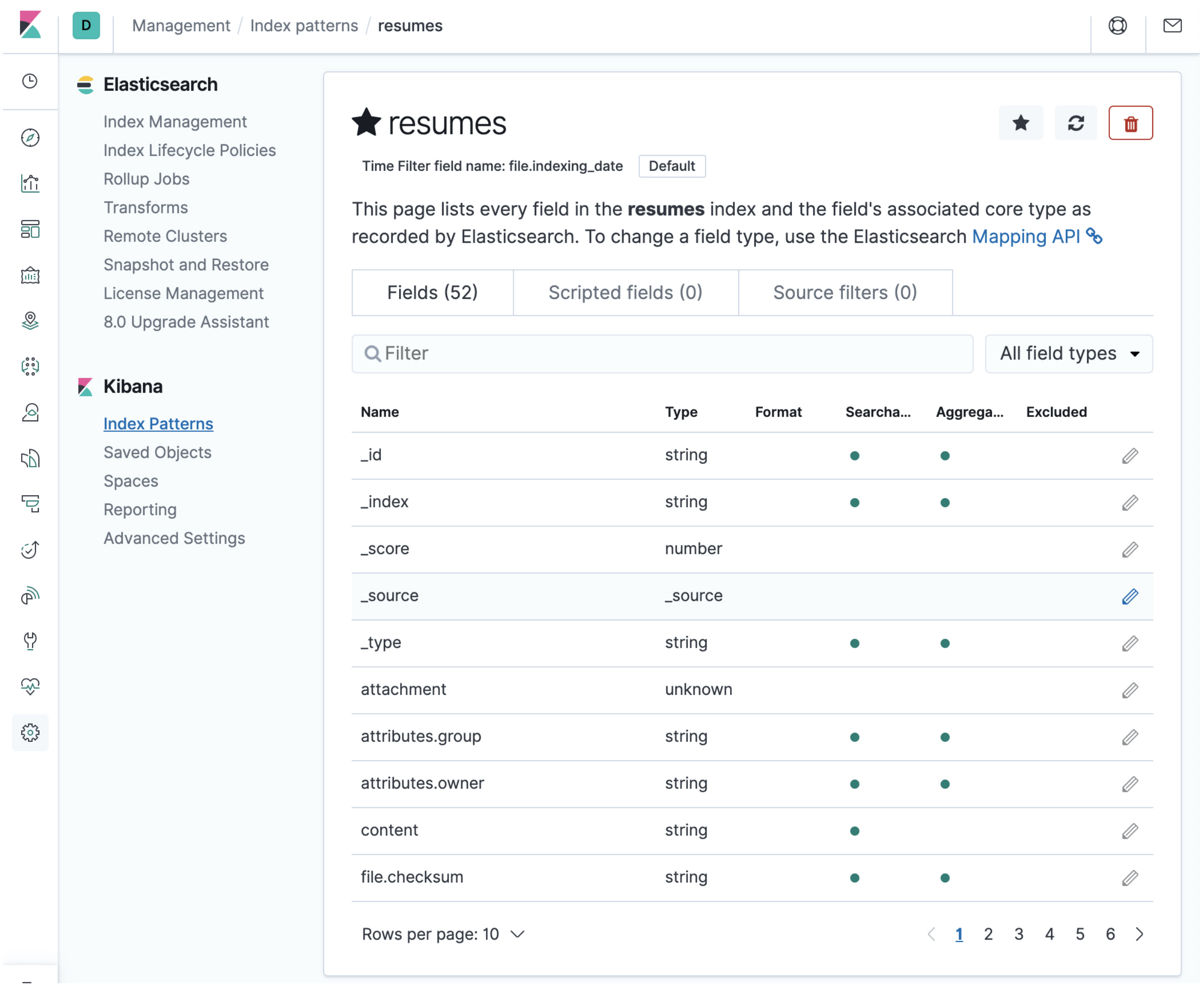
Time Filter field name Refresh

✓
file.created
file.indexing_date
file.last_accessed
file.last_modified
meta.created
meta.date
meta.metadata_date
meta.print_date
———
I don't want to use the Time Filter

Back

Create index pattern

- Click on "Create index pattern". You should see something like:



resumes

Time Filter field name: file.indexing_date Default

This page lists every field in the **resumes** index and the field's associated core type as recorded by Elasticsearch. To change a field type, use the Elasticsearch Mapping API

Fields (52) | Scripted fields (0) | Source filters (0)

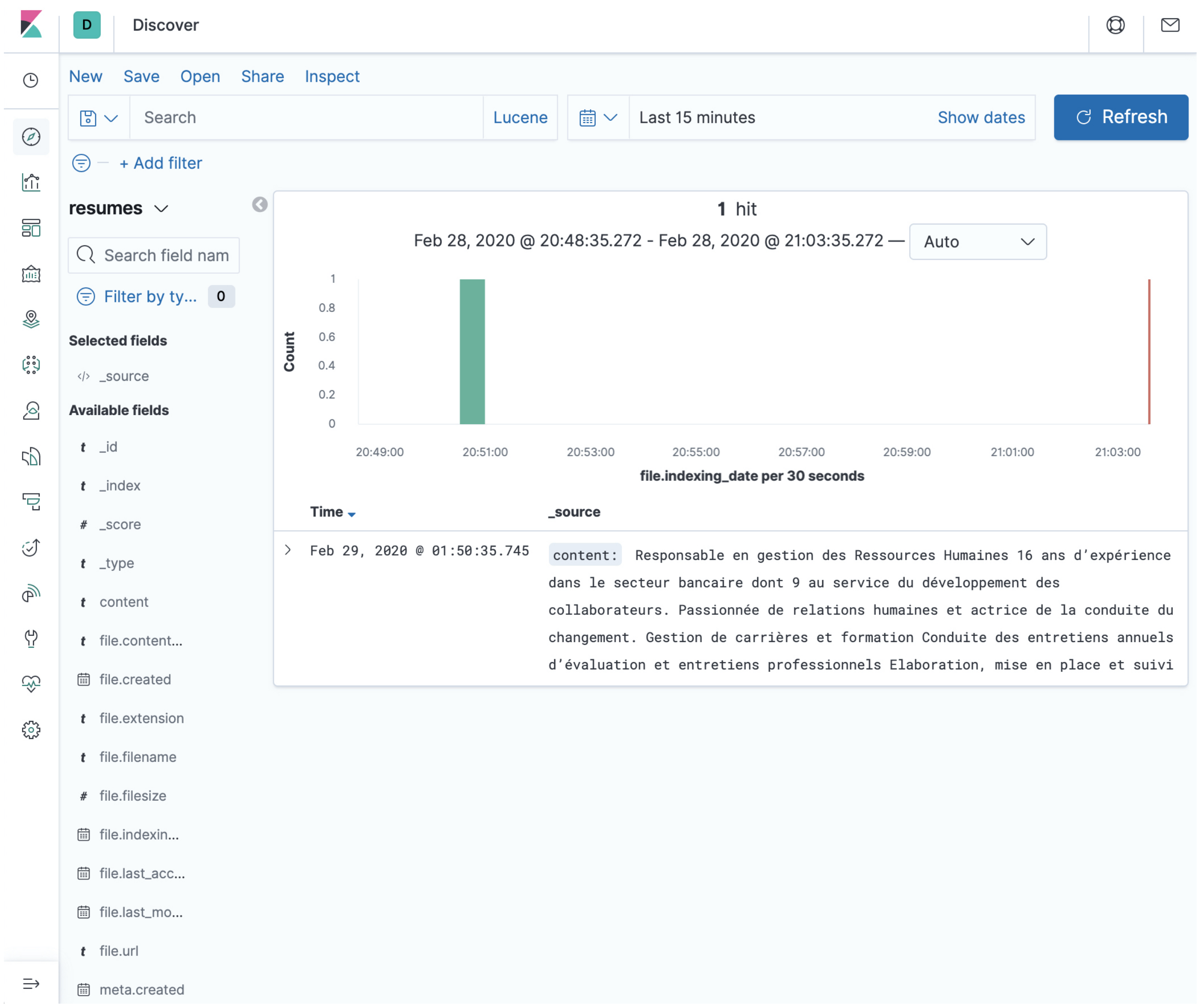
Filter | All field types

Name	Type	Format	Searcha...	Aggrega...	Excluded
_id	string		●	●	
_index	string		●	●	
_score	number				
_source	_source				
_type	string		●	●	
attachment	unknown				
attributes.group	string		●	●	
attributes.owner	string		●	●	
content	string		●		
file.checksum	string		●	●	

Rows per page: 10 | 1 2 3 4 5 6

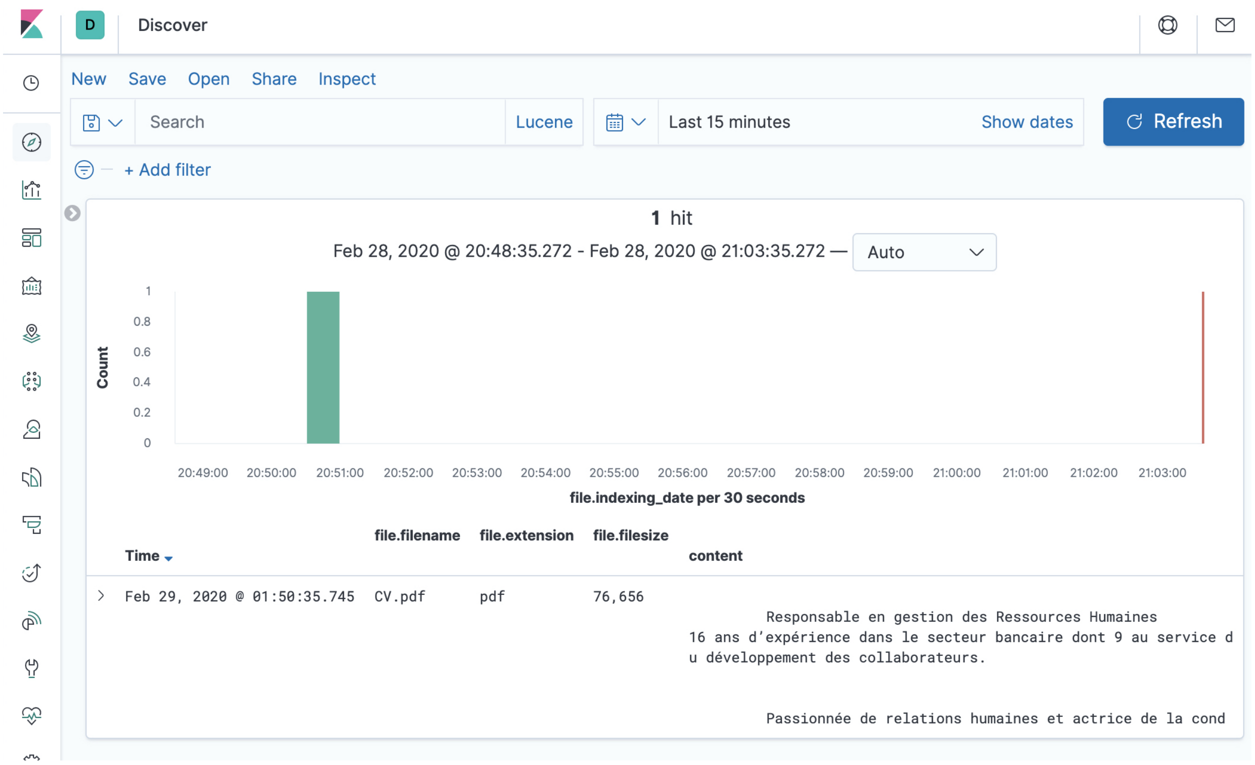
6.5 Search for the CVs

- Open Kibana
- Go to the Discover page
- Depending on the date you selected in the *Create Index pattern* step, you should see something similar to the following image. If you don't see it, you probably have to adjust the time picker to make sure you are looking at the right period of time.



```
Feb 29, 2020 @ 01:50:35.745   content: Responsable en gestion des Ressources Humaines 16 ans d'expérience
                              dans le secteur bancaire dont 9 au service du développement des
                              collaborateurs. Passionnée de relations humaines et actrice de la conduite du
                              changement. Gestion de carrières et formation Conduite des entretiens annuels
                              d'évaluation et entretiens professionnels Elaboration, mise en place et suivi
```

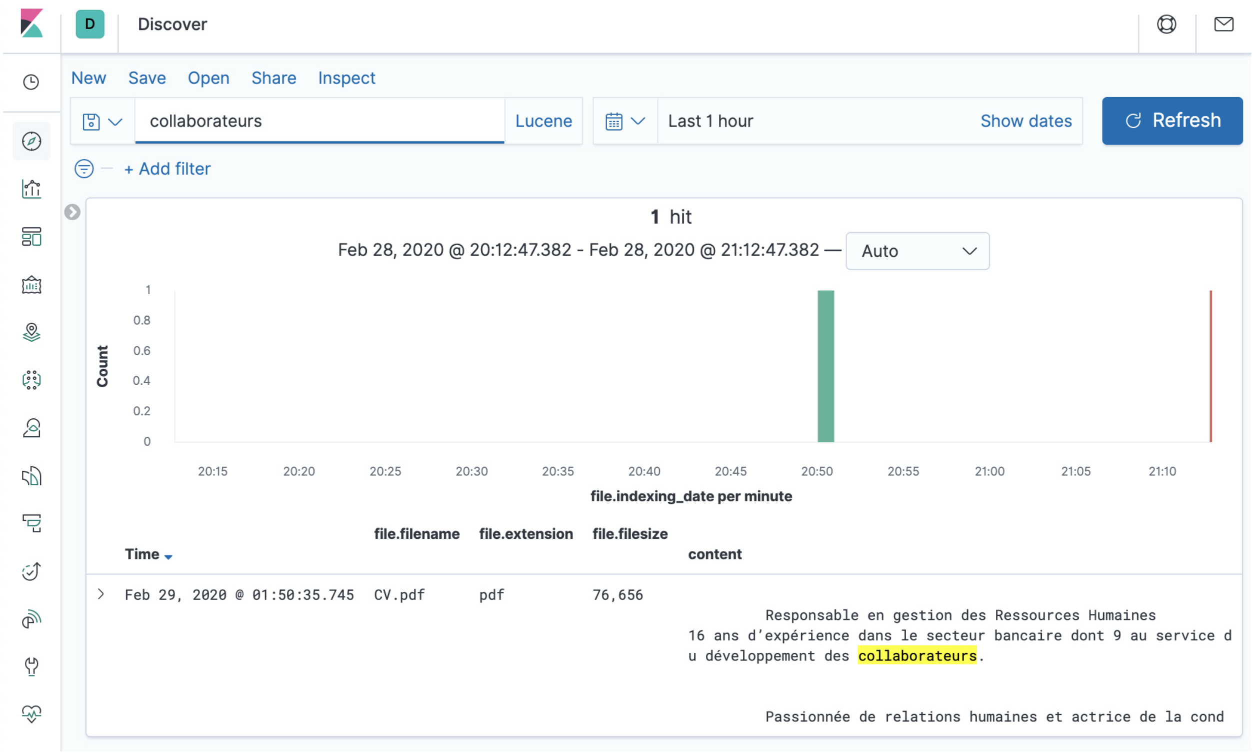
- You can select the fields you'd like to display in the result page, such as `content`, `file.filename`, `file.extension`, `file.url`, `file.filesize`, etc.


Discover
New Save Open Share Inspect
Search
Lucene
Last 15 minutes
Show dates
Refresh
+ Add filter
1 hit
Feb 28, 2020 @ 20:48:35.272 - Feb 28, 2020 @ 21:03:35.272 —
Auto
Count
file.indexing_date per 30 seconds

```
Time    file.filename  file.extension  file.filesize  content
Feb 29, 2020 @ 01:50:35.745  CV.pdf  pdf  76,656
          Responsable en gestion des Ressources Humaines
16 ans d'expérience dans le secteur bancaire dont 9 au service d
u développement des collaborateurs.

          Passionnée de relations humaines et actrice de la cond
```

- Of course, you can search for content, like `collaborateurs` here and see the highlighted content.


Discover
New Save Open Share Inspect
collaborateurs
Lucene
Last 1 hour
Show dates
Refresh
+ Add filter
1 hit
Feb 28, 2020 @ 20:12:47.382 - Feb 28, 2020 @ 21:12:47.382 —
Auto
Count
file.indexing_date per minute

```
Time    file.filename  file.extension  file.filesize  content
Feb 29, 2020 @ 01:50:35.745  CV.pdf  pdf  76,656
          Responsable en gestion des Ressources Humaines
16 ans d'expérience dans le secteur bancaire dont 9 au service d
u développement des collaborateurs.

          Passionnée de relations humaines et actrice de la cond
```

6.6 Adding new files

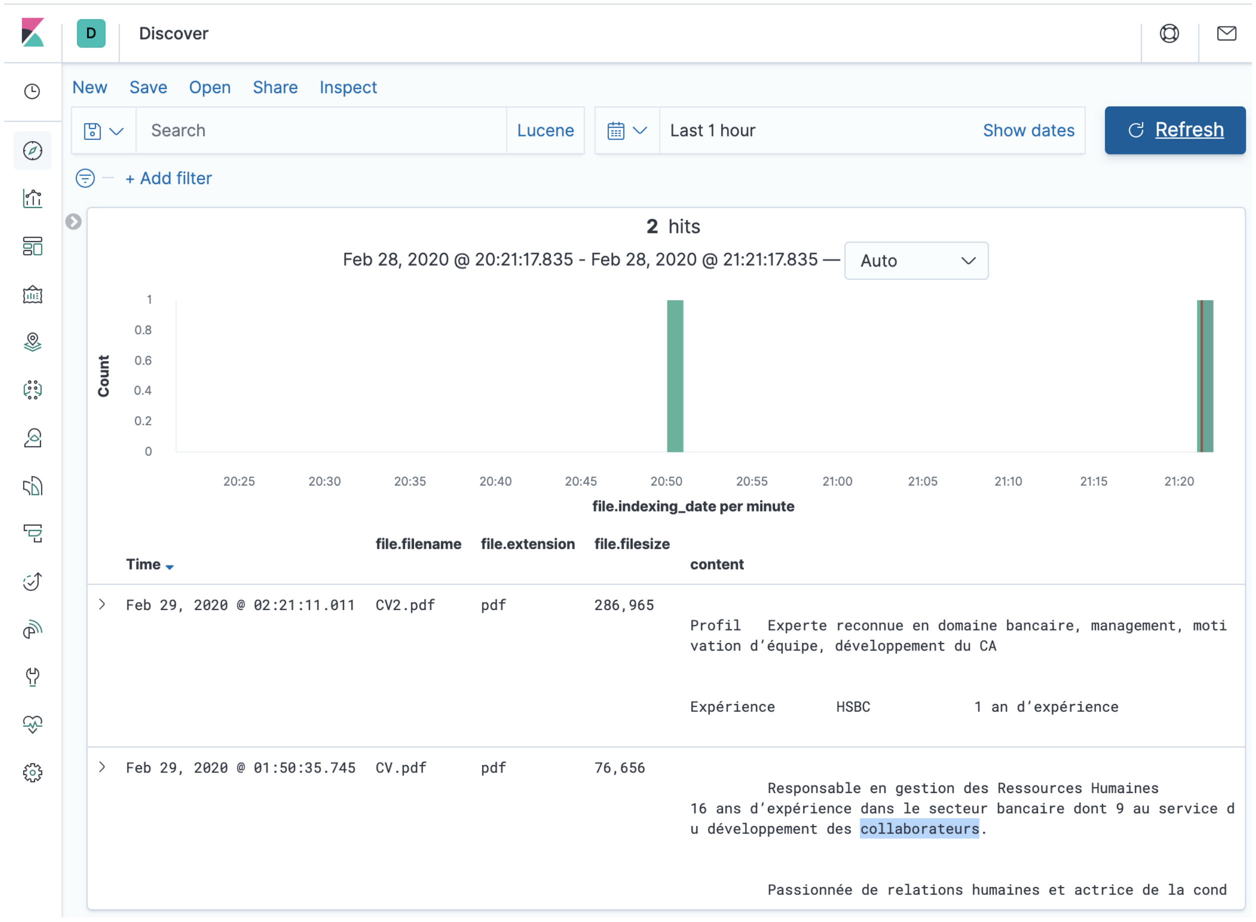
Just copy new files in the `resumes` folder. It could take up to 15 minutes for FSCrawler to detect the change. This is the default value for `update_rate` option. You can also change this value. See *Update rate*.

Note: On some OS, moving files won't touch the modified date and the "new" files won't be detected. It's then better probably to copy the files instead.

You might have to "touch" the files like:

```
touch /path/to/resumes/CV2.pdf
```

Just hit the Kibana refresh button and see the changes.



```
Time            file.filename  file.extension  file.filesize  content
> Feb 29, 2020 @ 02:21:11.011  CV2.pdf  pdf  286,965  Profil   Experte reconnue en domaine bancaire, management, moti
                                                      vation d'équipe, développement du CA

                                                      Expérience      HSBC          1 an d'expérience

> Feb 29, 2020 @ 01:50:35.745  CV.pdf   pdf  76,656          Responsable en gestion des Ressources Humaines
                                                      16 ans d'expérience dans le secteur bancaire dont 9 au service d
                                                      u développement des collaborateurs.

                                                             Passionnée de relations humaines et actrice de la cond
```

CHAPTER

SEVEN

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.yaml` 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*
- *FTP settings*
- *Elasticsearch settings*

CHAPTER

EIGHT

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".

8.1 OCR settings

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

Name	Default value	Documentation
`fs.ocr.enabled`	`true`	*Disable/Enable OCR*
`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*
`fs.ocr.pdf_strategy`	`ocr_and_text`	*OCR PDF Strategy*

8.2 Disable/Enable OCR

New in version 2.7.

You can completely disable using OCR by setting `fs.ocr.enabled` property in your `~/.fscrawler/test/_settings.yaml` file:

```
name: "test"
fs:
  url: "/path/to/data/dir"
  ocr:
    enabled: false
```

By default, OCR is activated if tesseract can be found on your system.

8.3 OCR Language

If you are using the default Docker image (see *Using docker*) or if you have installed any of the Tesseract Languages, you can use them when parsing your documents by setting `fs.ocr.language` property in your `~/.fscrawler/test/_settings.yaml` file:

```yaml
name: "test"
fs:
  url: "/path/to/data/dir"
  ocr:
    language: "eng"
```

Note: You can define multiple languages by using + sign as a separator:

```yaml
name: "test"
fs:
  url: "/path/to/data/dir"
  ocr:
    language: "eng+fas+fra"
```

8.4 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.yaml` file:

```yaml
name: "test"
fs:
  url: "/path/to/data/dir"
  ocr:
    path: "/path/to/tesseract/bin/"
```

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

8.5 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.yaml` file:

```yaml
name: "test"
fs:
  url: "/path/to/data/dir"
  ocr:
    path: "/path/to/tesseract/bin/"
    data_path: "/path/to/tesseract/share/tessdata/"
```

8.6 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.yaml` file:

```yaml
name: "test"
fs:
  url: "/path/to/data/dir"
  ocr:
    output_type: "hocr"
```

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

8.7 OCR PDF Strategy

By default, FSCrawler will also 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.ocr.pdf_strategy` to `"no_ocr"` or to `"auto"`:

```yaml
name: "test"
fs:
  ocr:
    pdf_strategy: "auto"
```

Supported strategies are:

- `auto`: No OCR is performed on PDF documents if there is more than 10 characters extracted. See PDFParser OCR Options.
- `no_ocr`: No OCR is performed on PDF documents. OCR might be performed on images though if OCR is not disabled. See *Disable/Enable OCR*.
- `ocr_only`: Only OCR is performed.
- `ocr_and_text`: OCR and text extraction is performed.

Note: When omitted, `ocr_and_text` value is used. If you have performance issues, it's worth using the `auto` option instead as only documents with barely no text will go through the OCR process.

CHAPTER

NINE

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.

Then 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"
}
```

To enable CORS (Cross-Origin Request Sharing) functionality you will need to set `enable_cors: true` in your job settings.

Read the *REST service* chapter for more information.

CHAPTER TEN

SUPPORTED FORMATS

FSCrawler supports all formats Tika supports, like:

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

CHAPTER

ELEVEN

TIPS AND TRICKS

11.1 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.

11.2 Workaround for huge temporary files

fscrawler uses a media library that currently does not clean up their temporary files. Parsing MP4 files may create very large temporary files in /tmp. The following commands could be useful e.g. as a cronjob to automatically delete those files once they are old and no longer in use. Adapt the commands as needed.

```
# Check all files in /tmp
find /tmp \( -name 'apache-tika-*.tmp-*' -o -name 'MediaDataBox*' \) -type f -mmin +15 !␣
↪-exec fuser -s {} \; -delete

# When using a systemd service with PrivateTMP enabled
find $(find /tmp -maxdepth 1 -type d -name 'systemd-private-*-fscrawler.service-*') \( -
↪name 'apache-tika-*.tmp-*' -o -name 'MediaDataBox*' \) -type f -mmin +15 ! -exec fuser␣
↪-s {} \; -delete
```

11.3 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.

11.4 Using docker with FSCrawler REST

To use the REST service available from 2.2 you can add the `--rest` flag to the FSCrawler docker container `command:`. Note that you must expose the same ports that the REST service opens on in the docker container. For example, if your REST service starts on `127.0.0.1:8080` then expose the same ports in your FSCrawler docker-compose image:

```
fscrawler:
  context: ${PWD}
  dockerfile: Dockerfile-fscrawler
container_name: fscrawler
restart: always
...
ports:
  - "8080:8080"
...
```

Then expose the docker container you've created by changing the IP of the REST URL in your `settings.yaml` to the docker-compose container name:

```
rest :
  url: "http://fscrawler:8080"
```

Pull the Docker image:

```
docker pull dadoonet/fscrawler
```

Run it:

```
docker run dadoonet/fscrawler job
```

CHAPTER

TWELVE

DIRECTORY LAYOUT

The directory layout of the project is as follows:

```
.
├── NOTICE
├── LICENSE
├── README.md
├── bin
│   ├── fscrawler
│   └── fscrawler.bat
├── config
│   ├── log4j2.xml
│   └── log4j2-file.xml
├── external
├── lib
└── logs
    ├── documents.log
    └── fscrawler.log
```

The `bin` directory contains the scripts to run FSCrawler.

The `lib` directory contains the FSCrawler jar file and all the dependencies.

New in version 2.10.

The `config` directory contains the configuration files. See *Configuring the logger*.

The `external` directory contains the external libraries you could add to FSCrawler. For example, if you want to add the `jai-imageio-jpeg2000` library to add support for JPEG2000 images, you can download it from Maven Central and put the `jai-imageio-jpeg2000-1.4.0.jar` file in the `external` directory.

As this directory is empty by default, you can also mount it when using Docker images:

```
docker run -it --rm \
     -v ~/.fscrawler:/root/.fscrawler \
     -v ~/tmp:/tmp/es:ro \
     -v "$PWD/external:/usr/share/fscrawler/external" \
     dadoonet/fscrawler fscrawler job_name
```

See also *Using docker* and *Using docker compose*.

The `logs` directory contains the log files. See *Configuring the logger*.

CHAPTER THIRTEEN

CLI OPTIONS

- `--help` displays help
- `--silent` runs in silent mode. No output is generated on the console.
- `--config_dir` defines directory where jobs are stored instead of default `~/.fscrawler`.
- `--api_key` defines the Elasticsearch Api Key to use. Do not use with `--username` or `--access_token`. Read *Using Credentials (Security)*.
- `--access_token` defines the Elasticsearch Access Token to use. Do not use with `--username` or `--api_key`. Read *Using Credentials (Security)*.
- `--username` defines the username to use (Deprecated). Do not use with `--api_key` or `--access_token`. Read *Using Credentials (Security)*.
- `--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*.

13.1 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`.

13.2 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.

13.3 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 FOURTEEN

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

FIFTEEN

CONFIGURING THE LOGGER

FSCrawler comes with a default logger configuration which can be found in the FSCrawler installation dir as `config/log4j2.xml` file.

You can modify it to suit your needs. It will be automatically reloaded every 30 seconds.

There are some properties to make your life easier to change the log levels or the log dir:

```
<Properties>
   <Property name="LOG_LEVEL">info</Property>
   <Property name="DOC_LEVEL">info</Property>
   <Property name="LOG_DIR">logs</Property>
</Properties>
```

You can control where FSCrawler will store the logs and the log levels by setting `LOG_DIR`, `LOG_LEVEL` and `DOC_LEVEL` Java properties.

```
FS_JAVA_OPTS="-DLOG_DIR=path/to/logs_dir -DLOG_LEVEL=trace -DDOC_LEVEL=debug" bin/
↪fscrawler
```

By default, it will log everything in the `logs` directory inside the installation folder.

Two log files are generated:

- One is used to log FSCrawler code execution, named `fscrawler.log`. It's automatically rotated every day or after 20mb of logs and gzipped. Logs are removed after 7 days.
- One is used to trace all information about documents, named `documents.log`. It's automatically rotated every day or after 20mb of logs and gzipped. Logs are removed after 7 days.

You can change this strategy by modifying the `config/log4j2.xml` file. Please read Log4J2 documentation on how to configure Log4J.

Note: FSCrawler detects automatically on Linux machines when it's running in background or foreground. When in background, the logger configuration file used is `config/log4j2-file.xml`.

CHAPTER SIXTEEN

STATUS FILES

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

- `~/.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

SEVENTEEN

EXAMPLE JOB FILE SPECIFICATION

The job file (`~/.fscrawler/test/_settings.yaml`) for the job name `test` must comply to the following `yaml` specifications:

```yaml
# required
name: "test"

# required
fs:

  # define a "local" file path crawler, if running inside a docker container this must␣
↪be the path INSIDE the container
  url: "/path/to/docs"
  follow_symlink: false
  remove_deleted: true
  continue_on_error: false

  # scan every 5 minutes for changes in url defined above
  update_rate: "5m"

  # opional: define includes and excludes, "~" files are excluded by default if not␣
↪defined below
  includes:
  - "*.doc"
  - "*.xls"
  excludes:
  - "resume.doc"

  # optional: do not send big files to TIKA
  ignore_above: "512mb"

  # special handling of JSON files, should only be used if ALL files are JSON
  json_support: false
  add_as_inner_object: false

  # special handling of XML files, should only be used if ALL files are XML
  xml_support: false

  # use MD5 from filename (instead of filename) if set to false
  filename_as_id: true
```

(continues on next page)

(continued from previous page)

```yaml
  # include size ot file in index
  add_filesize: true

  # inlcude user/group of file only if needed
  attributes_support: false

  # do you REALLY want to store every file as a copy in the index ? Then set this to true
  store_source: false

  # you may want to store (partial) content of the file (see indexed_chars)
  index_content: true

  # how much data from the content of the file should be indexed (and stored inside the␣
↪index), set to 0 if you need checksum, but no content at all to be indexed
  #indexed_chars: "0"
  indexed_chars: "10000.0"

  # usually file metadata will be stored in separate fields, if you want to keep the␣
↪original set, set this to true
  raw_metadata: false

  # optional: add checksum meta (requires index_content to be set to true)
  checksum: "MD5"

  # recommmended, but will create another index
  index_folders: true

  lang_detect: false

  ocr.pdf_strategy: noocr
  #ocr:
  #  language: "eng"
  #  path: "/path/to/tesseract/if/not/available/in/PATH"
  #  data_path: "/path/to/tesseract/tessdata/if/needed"

# optional: only required if you want to SSH to another server to index documents from␣
↪there
server:
  hostname: "localhost"
  port: 22
  username: "dadoonet"
  password: "password"
  protocol: "SSH"
  pem_path: "/path/to/pemfile"

# required
elasticsearch:
  nodes:
  # With Cloud ID
  - cloud_id: "CLOUD_ID"
  # With URL
  - url: "http://127.0.0.1:9200"
```

(continues on next page)

(continued from previous page)

```yaml
  bulk_size: 1000
  flush_interval: "5s"
  byte_size: "10mb"
  # choose one of the 3 following options:
  # 1 - Using access token
  access_token:
↪"dGhpcyBpcyBub3QgYSByZWFsIHRva2VuIGJ1dCBpdCBpcyBvbmx5IHRlc3QgZGF0YS4gZG8gbm90IHRyeSB0byByZWFkIHRva2VuI
↪"
  # 2 - Using Api Key
  api_key: "VnVhQ2ZHY0JDZGJrUW0tZTVhT3g6dWkybHAyYXhUTm1zeWFrdzl0dk5udw=="
  # 3 - Using username/password (not recommended / deprecated)
  username: "elastic"
  password: "password"
  # optional, defaults to ``name``-property
  index: "test_docs"
  # optional, defaults to "test_folders", used when es.index_folders is set to true
  index_folder: "test_fold"
rest:
  # only is started with --rest option
  url: "http://127.0.0.1:8080/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*

New in version 2.7.

You can define your job settings either in `_settings.yaml` (using `.yaml` extension) or in `_settings.json` (using `.json` extension).

CHAPTER

EIGHTEEN

THE MOST SIMPLE CRAWLER

You can define the most simple crawler job by writing a `~/.fscrawler/test/_settings.yaml` 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

NINETEEN

LOCAL FS SETTINGS

Contents

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`	`false`	*Enabling 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.ocr.pdf_strategy`	`ocr_and_text`	*OCR integration*
`fs.indexed_chars`	`100000.0`	*Extracted characters*
`fs.ignore_above`	`null`	*Ignore above*
`fs.checksum`	`false`	*File Checksum*
`fs.follow_symlinks`	`false`	*Follow Symlinks*
`fs.tika_config_path`	`null`	*Tika Config Path*

19.1 Root directory

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

```yaml
name: "test"
fs:
  url: "/path/to/data/dir"
```

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

19.2 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:

```yaml
name: "test"
fs:
  update_rate: "15m"
```

Or a 3 hours update rate:

```yaml
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.

19.3 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.yaml` file:

```yaml
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.yaml` file:

```yaml
name: "test"
fs:
  excludes:
  - "/folderB/subfolder*"
```

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

Since the includes and excludes work on the entire *path of the file* you must consider that when using wildcards. Below are some includes and excludes pattern to help convey the idea better.

Pattern	Includes	Excludes
`*.jpg`	Include all jpg files	exclude all jpg files
`/images/*.jpg`	Include all jpg files in the images directory	Exclude all jpg files in the images directory
`*/old-*.jpg`	Include all jpg files that start with `old-`	Exclude all jpg files that start with `old-`

New in version 2.6.

If a folder contains a file named `.fscrawlerignore`, this folder and its subfolders will be entirely skipped.

19.4 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.yaml` file:

```yaml
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.

19.5 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`.

```yaml
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.

19.6 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.

19.7 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
```

19.8 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.

See `elasticsearch.index_folder` below for the name of the index to be used to store the folder data (if `es.index_folders` is set to `true`).

```
name: "test"
fs:
  index_folders: false
```

19.9 Dealing with multiple types and multiple dirs

If you have more than one type, create as many crawlers as types and/or folders:

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

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

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

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

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

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

19.10 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.yaml`:

```yaml
name: "test_type1"
fs:
  url: "/tmp"
  includes:
  - "type1*.json"
  json_support: true
elasticsearch:
  index: "mydocs1"
  index_folder: "myfolders1"
```

`~/.fscrawler/test_type2.yaml`:

```yaml
name: "test_type2"
fs:
  url: "/tmp"
  includes:
  - "type2*.json"
  json_support: true
elasticsearch:
  index: "mydocs2"
  index_folder: "myfolders2"
```

`~/.fscrawler/test_type3.yaml`:

```yaml
name: "test_type3"
fs:
  url: "/tmp"
  includes:
  - "*.xml"
  xml_support: true
elasticsearch:
  index: "mydocs3"
  index_folder: "myfolders3"
```

19.11 Using filename as elasticsearch `_id`

Please note that the document `_id` is generated as a 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:

```yaml
name: "test"
fs:
  filename_as_id: true
```

19.12 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`.

```yaml
name: "test"
fs:
  attributes_support: true
```

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

19.13 Enabling raw metadata

FSCrawler can extract all found metadata within a `meta.raw` object in addition to the standard metadata fields. If you want to enable this feature, you can set `raw_metadata` to `true`.

```
name: "test"
fs:
  raw_metadata: true
```

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",
   "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`.

Note: Note that if you have a lot of different type of files, that can generate a lot of raw metadata which can make you hit the total number of field limit in elasticsearch mappings. In which case you will need to change the index settings `foo`.

See elasticsearch documentation

19.14 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:

```yaml
name: "test"
fs:
  add_filesize: false
```

19.15 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`):

```yaml
name: "test"
fs:
  remove_deleted: false
```

Note: Setting `remove_deleted` is forced to `false` when using the Workplace Search output (wpsearch-settings).

19.16 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`):

```yaml
name: "test"
fs:
  index_content: false
```

19.17 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`):

```yaml
name: "test"
fs:
  continue_on_error: true
```

19.18 Language detection

New in version 2.2.

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

```yaml
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:

```yaml
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.

19.19 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
    }
  }
}
```

19.20 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!

19.21 Ignore Above

New in version 2.5.

By default (if `index_content` set to `true`) FSCrawler will send every single file to Tika, 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: "512mb"
```

19.22 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`.

Note: You MUST set `index_content` to true to allow this feature to work. Nevertheless you MAY set `indexed_chars` to 0 if you do not need any content in the index.

You MUST NOT set `json_support` or `xml_support` to allow this feature to work also.

```
name: "test"
fs:
   # required
  index_content: true
  #indexed_chars: 0
  checksum: "MD5"
```

19.23 Follow Symlinks

New in version 2.7.

If you want FSCrawler to follow the symbolic links, you need to be explicit about it and set `follow_symlink` to `true`. Starting from version 2.7, symbolic links are not followed anymore.

```
name: "test"
fs:
  follow_symlink: true
```

19.24 Tika Config Path

New in version 2.10.

If you want to override the default tika parser configuration, you can set the path to a custom tika configuration file, which will be used instead.

```yaml
name: "test"
fs:
  tika_config_path: '/path/to/tikaConfig.xml'
```

An example tika config file is shown below. See Configuring Tika for more information.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<properties>
  <service-loader dynamic="true"/>
  <service-loader loadErrorHandler="IGNORE"/>
  <parsers>
    <!-- Use Default Parser for files, but Default Parser will never use HTML parser -->
    <parser class="org.apache.tika.parser.DefaultParser">
      <parser-exclude class="org.apache.tika.parser.html.HtmlParser"/>
    </parser>
    <!-- Use a different parser for XHTML -->
    <parser class="org.apache.tika.parser.xml.XMLParser">
      <mime>application/xhtml+xml</mime>
    </parser>
  </parsers>
</properties>
```

CHAPTER

TWENTY

SSH SETTINGS

You can index files remotely using SSH.

Contents

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*

20.1 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"
server:
```

(continues on next page)

(continued from previous page)

```
  hostname: "mynode.mydomain.com"
  port: 22
  username: "username"
  password: "password"
  protocol: "ssh"
```

20.2 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"
  password: "password"
  protocol: "ssh"
  pem_path: "/path/to/private_key.pem"
```

20.3 Windows drives

When using Windows, you might want to index documents coming from another drive than `C:`. To specify the drive, you need to use the following format:

```
name: "test"
fs:
  url: "/D:/path/to/data/dir/on/server"
server:
  hostname: "mynode.mydomain.com"
  port: 22
  username: "username"
  password: "password"
  protocol: "ssh"
```

CHAPTER

TWENTYONE

FTP SETTINGS

You can index files remotely using FTP.

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

Name	Default value	Documentation
`server.hostname`	`null`	Hostname
`server.port`	`21`	Port
`server.username`	`anonymous`	*Username / Password*
`server.password`	`null`	*Username / Password*
`server.protocol`	`"local"`	Set it to `ftp`

21.1 Username / Password

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

- FS URL: `/path/to/data/dir/on/server`
- Server: `mynode.mydomain.com`
- Username: `username` (default to `anonymous`)
- Password: `password`
- Protocol: `ftp` (default to `local`)
- Port: `21` (default to `21`)

```
name: "test"
fs:
  url: "/path/to/data/dir/on/server"
server:
  hostname: "mynode.mydomain.com"
  port: 21
  username: "username"
  password: "password"
  protocol: "ftp"
```

CHAPTER

TWENTYTWO

ELASTICSEARCH SETTINGS

Contents

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.push_templates	true	*Mappings*
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.path_prefix	null	*Path prefix*
elasticsearch.api_key	null	*API Key*
elasticsearch.access_token	null	*Access Token*
elasticsearch.username	null	*Using Credentials (Security)*
elasticsearch.password	null	*Using Credentials (Security)*
elasticsearch.ssl_verification	true	*Using Credentials (Security)*

22.1 Index settings

22.1.1 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). You can change it by setting `index` field:

```yaml
name: "test"
elasticsearch:
  index: "docs"
```

22.1.2 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:

```yaml
name: "test"
elasticsearch:
  index_folder: "folders"
```

22.1.3 Mappings

New in version 2.10.

FSCrawler defines the following Component Templates to define the index settings and mappings:

- `fscrawler_alias`: defines the alias `fscrawler` so you can search using this alias.
- `fscrawler_settings_shards`: defines the number of shards to use for the index.
- `fscrawler_settings_total_fields`: defines the maximum number of fields for the index.
- `fscrawler_mapping_attributes`: defines the mapping for the `attributes` field.
- `fscrawler_mapping_file`: defines the mapping for the `file` field.

- `fscrawler_mapping_path`: defines an define an analyzer named `fscrawler_path` which uses a path hierarchy tokenizer and the mapping for the `path` field.
- `fscrawler_mapping_attachment`: defines the mapping for the `attachment` field.
- `fscrawler_mapping_content`: defines the mapping for the `content` field.
- `fscrawler_mapping_meta`: defines the mapping for the `meta` field.

You can see the content of those templates by running:

```
GET _component_template/fscrawler*
```

Then, FSCrawler applies those templates to the indices being created.

You can stop FSCrawler creating/updating the index templates for you by setting `push_templates` to `false`:

```
name: "test"
elasticsearch:
  push_templates: false
```

If you want to know what are the component templates and index templates that will be created, you can get them from the source.

Creating your own mapping (analyzers)

If you want to define your own index settings and mapping to set analyzers for example, you can update the needed component template **before starting the FSCrawler**.

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

```
PUT _component_template/fscrawler_mapping_content
{
  "template": {
    "mappings": {
      "properties": {
        "content": {
          "type": "text",
          "analyzer": "french"
        }
      }
    }
  }
}
```

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.

22.2 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.

22.3 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.

22.4 Node settings

FSCrawler is using elasticsearch REST layer to send data to your running cluster. By default, it connects to `https://127.0.0.1:9200` which is the default when running a local node on your machine. Note that using `https` requires SSL Configuration set up. For more information, read *SSL Configuration*.

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

```yaml
name: "test"
elasticsearch:
  nodes:
  - url: "https://mynode1.mycompany.com:9200"
```

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:

```yaml
name: "test"
elasticsearch:
  nodes:
  - cloud_id:
↪"fscrawler:ZXVyb3BlLXdlc3QxLmdjcC5jbG91ZC5lcy5pbyQxZDFlYTk5Njg4Nzc0NWE2YTJiN2NiNzkzMTUzNDhhMyQyOTk1MD
↪"
```

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 (Security)*.

You can define multiple nodes:

```yaml
name: "test"
elasticsearch:
  nodes:
  - url: "https://mynode1.mycompany.com:9200"
  - url: "https://mynode2.mycompany.com:9200"
  - url: "https://mynode3.mycompany.com:9200"
```

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

```yaml
name: "test"
elasticsearch:
  nodes:
  - url: "https://CLUSTERID.eu-west-1.aws.found.io:9243"
```

For more information, read *SSL Configuration*.

22.5 Path prefix

New in version 2.7: If your elasticsearch is running behind a proxy with url rewriting, you might have to specify a path prefix. This can be done with `path_prefix` setting:

```
name: "test"
elasticsearch:
  nodes:
  - url: "http://mynode1.mycompany.com:9200"
  path_prefix: "/path/to/elasticsearch"
```

Note: The same `path_prefix` applies to all nodes.

22.6 Using Credentials (Security)

If you have a secured cluster, you can use several methods to connect to it:

- API Key
- Access Token
- Basic Authentication (not recommended / deprecated)

22.6.1 API Key

New in version 2.10.

Let's create an API Key named `fscrawler`:

```
POST /_security/api_key
{
  "name": "fscrawler"
}
```

This gives something like:

```
{
  "id": "VuaCfGcBCdbkQm-e5aOx",
  "name": "fscrawler",
  "expiration": 1544068612110,
  "api_key": "ui2lp2axTNmsyakw9tvNnw",
  "encoded": "VnVhQ2ZHY0JDZGJrUW0tZTVhT3g6dWkybHAyYXhUTm1zeWFrdzl0dk5udw=="
}
```

Then you can use the encoded API Key in FSCrawler settings:

```
name: "test"
elasticsearch:
  api_key: "VnVhQ2ZHY0JDZGJrUW0tZTVhT3g6dWkybHAyYXhUTm1zeWFrdzl0dk5udw=="
```

22.6.2 Access Token

New in version 2.10.

Let's create an API Key named `fscrawler`:

```
POST /_security/oauth2/token
{
  "grant_type" : "client_credentials"
}
```

This gives something like:

```
{
  "access_token" :
↪"dGhpcyBpcyBub3QgYSByZWFsIHRva2VuIGJ1dCBpdCBpcyBvbmx5IHRlc3QgZGF0YS4gZG8gbm90IHRyeSB0byByZWFkIHRva2Vu
↪",
  "type" : "Bearer",
  "expires_in" : 1200,
  "authentication" : {
    "username" : "test_admin",
    "roles" : [
      "superuser"
    ],
    "full_name" : null,
    "email" : null,
    "metadata" : { },
    "enabled" : true,
    "authentication_realm" : {
      "name" : "file",
      "type" : "file"
    },
    "lookup_realm" : {
      "name" : "file",
      "type" : "file"
    },
    "authentication_type" : "realm"
  }
}
```

Then you can use the generated Access Token in FSCrawler settings:

```
name: "test"
elasticsearch:
  access_token:
↪"dGhpcyBpcyBub3QgYSByZWFsIHRva2VuIGJ1dCBpdCBpcyBvbmx5IHRlc3QgZGF0YS4gZG8gbm90IHRyeSB0byByZWFkIHRva2Vu
↪"
```

22.6.3 Basic Authentication (deprecated)

New in version 2.2.

The best practice is to use *API Key* or *Access Token*. But if you have no other choice, you can still use Basic Authentication.

You can provide the `username` and `password` to FSCrawler:

```
name: "test"
elasticsearch:
  username: "elastic"
  password: "changeme"
```

Warning: Be aware that 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:
```

22.6.4 User permissions

If you want to use another user than the default `elastic` (which is admin), you will need to give him some permissions:

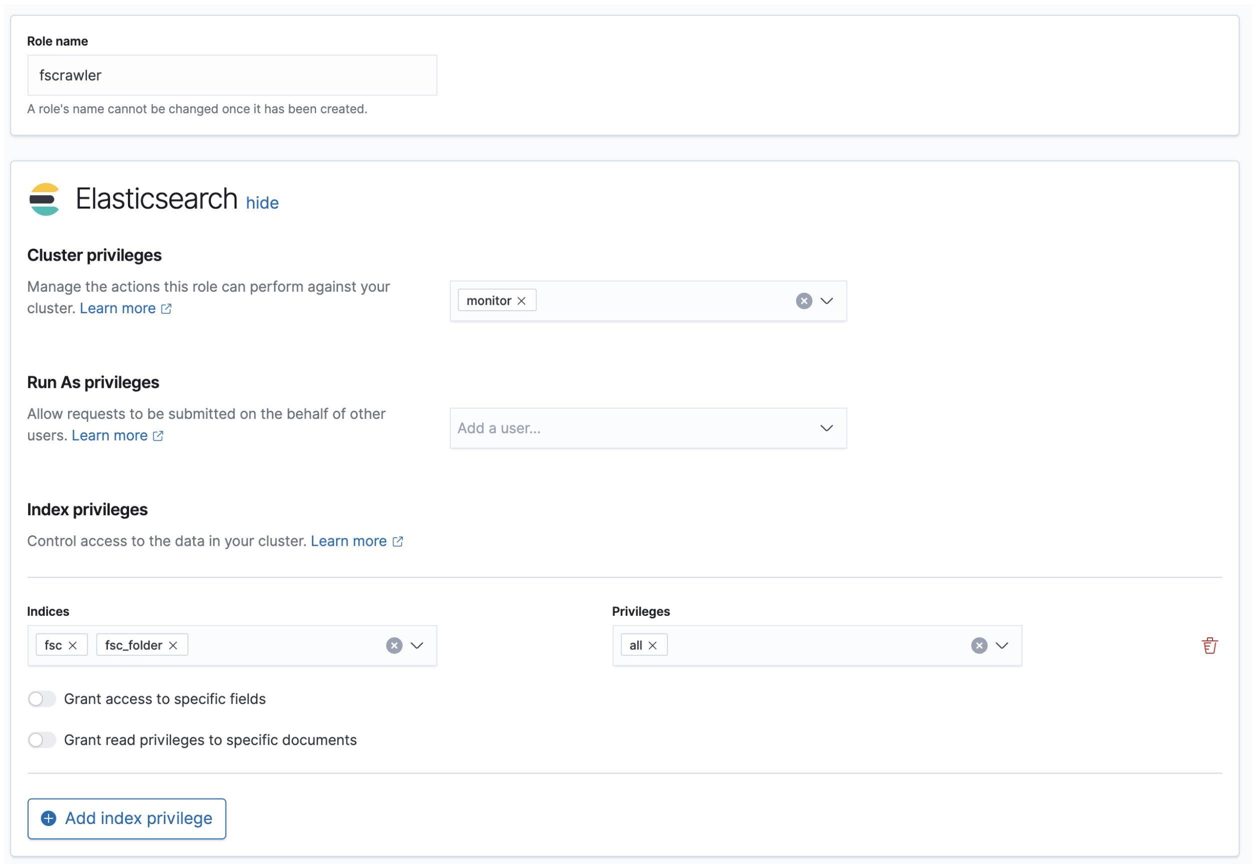
- `cluster:monitor`
- `indices:fsc/all`
- `indices:fsc_folder/all`

where `fsc` is the FSCrawler index name as defined in *Index settings for documents*.

This can be done by defining the following role:

```
PUT /_security/role/fscrawler
{
  "cluster" : [ "monitor" ],
  "indices" : [ {
      "names" : [ "fsc", "fsc_folder" ],
      "privileges" : [ "all" ]
  } ]
}
```

This also can be done using the Kibana Stack Management Interface.


Role name

fscrawler

A role's name cannot be changed once it has been created.

Elasticsearch hide

Cluster privileges

Manage the actions this role can perform against your cluster. Learn more

monitor

Run As privileges

Allow requests to be submitted on the behalf of other users. Learn more

Add a user...

Index privileges

Control access to the data in your cluster. Learn more

Indices	Privileges
fsc, fsc_folder	all

Grant access to specific fields

Grant read privileges to specific documents

Add index privilege

Then, you can assign this role to the user who will be defined within the `username` setting.

22.7 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. Certificate verification can be disabled by option `ssl_verification: false`

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:
```

(continues on next page)

(continued from previous page)

```
  nodes:
  - url: "https://localhost:9243"
```

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

22.8 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"
   }
}
```

22.9 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" : {
        "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 TWENTYTHREE

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.

Contents

23.1 General settings

New in version 2.10.

For all the APIs on this page, you can pass parameters in different ways.

You can use a query string parameter:

```
curl "http://127.0.0.1:8080/fscrawler/API?param1=foo&param2=bar"
```

You can use a header parameter:

```
curl -H "param1=foo" -H "param2=bar" "http://127.0.0.1:8080/fscrawler/API"
```

The rest of this documentation will assume using a query string parameter unless stated otherwise.

23.2 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,
      "lang_detect" : false
    },
    "elasticsearch" : {
      "nodes" : [ {
        "url" : "http://127.0.0.1:9200"
      } ],
      "index" : "fscrawler-rest-tests_doc",
      "index_folder" : "fscrawler-rest-tests_folder",
      "bulk_size" : 100,
      "flush_interval" : "5s",
      "byte_size" : "10mb",
      "username" : "elastic"
    },
    "rest" : {
      "url" : "http://127.0.0.1:8080/fscrawler",
      "enable_cors": false
    }
  }
}
```

23.3 Uploading a binary document

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

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

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" : {
    "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/_document?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"
    }
  }
}
```

23.4 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/_document?debug=true&
↪simulate=true"
```

23.5 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` as a parameter:

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

You can pass the `id` parameter within 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/_document"
```

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.

23.6 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 /_document` 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/_document"
```

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": [
      "Mon",
      "Tue",
      "Wed",
      "Thu",
```

(continues on next page)

(continued from previous page)

```
      "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.

23.7 Remove a document

New in version 2.10.

To remove a document, you can call `DELETE /_document` endpoint.

If you only know the filename, you can pass it to FSCrawler using the `filename` field:

```
curl -X DELETE "http://127.0.0.1:8080/fscrawler/_document?filename=test.txt"
```

It will give you a response similar to:

```
{
  "ok": true,
  "filename": "test.txt",
  "index": "rest",
  "id": "dd18bf3a8ea2a3e53e2661c7fb53534"
}
```

If you know the document id, you can pass it to FSCrawler within the url:

```
curl -X DELETE "http://127.0.0.1:8080/fscrawler/_document/dd18bf3a8ea2a3e53e2661c7fb53534
↪"
```

If the document does not exist, you will get the following response:

```
{
  "ok": false,
  "message": "Can not remove document [rest/test.txt]: Can not remove document rest/
↪dd18bf3a8ea2a3e53e2661c7fb53534 cause: NOT_FOUND",
  "filename": "test.txt",
```

(continues on next page)

(continued from previous page)

```
  "index": "rest",
  "id": "dd18bf3a8ea2a3e53e2661c7fb53534"
}
```

23.8 Specifying an elasticsearch index

By default, fscrawler creates document in the index defined in the `_settings.yaml` file. However, using the REST service, it is possible to require fscrawler to use different indexes, by setting the `index` parameter:

```
echo "This is my text" > test.txt
curl -F "file=@test.txt" "http://127.0.0.1:8080/fscrawler/_document?index=my-index"
curl -X DELETE "http://127.0.0.1:8080/fscrawler/_document?filename=test.txt&index=my-
↪index"
```

When uploading, you can pass the `id` parameter within the form data:

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

23.9 Enabling CORS

To enable Cross-Origin Request Sharing you will need to set `enable_cors:  true` under `rest` in your job settings. Doing so will enable the relevant access headers on all REST service resource responses (for example `/fscrawler` and `/fscrawler/_document`).

You can check if CORS is enabled with:

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

The response header should contain `Access-Control-Allow-*` parameters like:

```
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: origin, content-type, accept, authorization
Access-Control-Allow-Credentials: true
Access-Control-Allow-Methods: GET, POST, PUT, PATCH, DELETE, OPTIONS, HEAD
```

23.10 REST settings

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

Name	Default value	Documentation
`rest.url`	`http://127.0.0.1:8080/fscrawler`	Rest Service URL
`rest.enable_cors`	`false`	Enables or disables Cross-Origin Resource Sharing globally for all resources

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:

```yaml
name: "test"
rest:
  url: "http://192.168.0.1:8180/my_fscrawler"
```

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

CHAPTER

TWENTYFOUR

BUILDING THE PROJECT

This project is built with Maven. It needs Java >= 1.11. Source code is available on GitHub. Thanks to JetBrains for the IntelliJ IDEA License!

Contents

24.1 Clone the project

Use git to clone the project locally:

```
git clone git@github.com:dadoonet/fscrawler.git
cd fscrawler
```

24.2 Build the artifact

To build the project, run:

```
mvn clean package
```

The final artifacts are available in `distribution/esX/target` directory where `X` is the elasticsearch major version target.

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

```
mvn clean package -DskipTests
```

24.3 Integration tests

When running from the command line with `mvn` integration tests are ran against all supported versions. This is done by running a Docker instance of elasticsearch using the expected version.

A HTTP server is also started on port 8080 during the integration tests, alternatively the assigned port can be set with *-Dtests.rest.port=8090* argument.

24.3.1 Run tests from your IDE

To run integration tests from your IDE, you need to start tests in `fscrawler-it-common` module. But you need first to specify the Maven profile to use and rebuild the project.

- `es-7x` for Elasticsearch 7.x
- `es-6x` for Elasticsearch 6.x

24.3.2 Run a specific test from your Terminal

To run a specific integration test, just run:

```
mvn verify -am -Dtests.class=fr.pilato.elasticsearch.crawler.fs.test.integration.CLASS_
↪NAME -Dtests.method="METHOD_NAME"
```

24.3.3 Run tests with an external cluster

Launching the docker containers might take some time so if to want to run the test suite against an already running cluster, you need to provide a `tests.cluster.url` value. This will skip launching the docker instances.

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

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 -Dtests.cluster.url=http:/
↪/localhost:9200
```

Tip: If you want to run against a version 6, run:

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v6 -Dtests.cluster.url=http:/
↪/localhost:9200
```

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

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.cluster.url=http://127.0.0.1:9200 \
```

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

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.cluster.url=https://XYZ.es.io:9243
```

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

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.cluster.cloud_
↪id=fscrawler:ZXVyb3BlLXdlc3QxLmdjcC5jbG91ZC5lcy5pbyQxZDFlYTk5Njg4Nzc0NWE2YTJiN2NiNzkzMTUzNDhhMyQyOTk1
```

24.3.4 Using security feature

Integration tests are run by default against a secured Elasticsearch cluster.

New in version 2.7.

Secured tests are using by default `changeme` as the password. You can change this by using `tests.cluster.pass` option:

```
mvn verify -Dtests.cluster.pass=mystrongpassword
```

24.3.5 Testing Workplace Search connector

New in version 2.7.

The Workplace Search integration is automatically tested when running the integration tests. The maven process will start both elasticsearch and enterprise search nodes. Note that this could take several minutes before to have it up and running.

To test the Workplace Search connector against an existing cluster, you can provide the `tests.cluster.url` setting. This will skip launching the containers and all the test suite will run against this external cluster:

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 \
    -Dtests.cluster.url=http://localhost:9200 \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.workplace.url=http://localhost:3002
```

Note: By default, `tests.workplace.user` and `tests.workplace.pass` are using the same values as for `tests.cluster.user` and `tests.cluster.pass`. But if you want to use another username and password to connect to workplace search, you can override the settings:

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 \
    -Dtests.cluster.url=http://localhost:9200 \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.workplace.url=http://localhost:3002
    -Dtests.workplace.user=enterprise_search \
    -Dtests.workplace.pass=changeme
```

To run Workplace Search tests against the Enterprise Search service by Elastic, you can also use something like:

```
mvn verify -pl fr.pilato.elasticsearch.crawler:fscrawler-it-v7 \
    -Dtests.cluster.url=https://ALIAS.es.eu-west-3.aws.elastic-cloud.com:9243 \
    -Dtests.cluster.user=elastic \
    -Dtests.cluster.pass=changeme \
    -Dtests.workplace.url=https://ALIAS.ent.eu-west-3.aws.elastic-cloud.com \
    -Dtests.workplace.user=enterprise_search \
    -Dtests.workplace.pass=changeme
```

24.3.6 Changing the REST port

By default, FS crawler will run the integration tests using port `8080` for the REST service. You can change this by using `tests.rest.port` option:

```
mvn verify -Dtests.rest.port=8280
```

24.3.7 Randomized testing

FS Crawler uses the randomized testing framework. In case of failure, it will print a line like:

```
REPRODUCE WITH:
mvn test -Dtests.seed=AC6992149EB4B547 -Dtests.class=fr.pilato.elasticsearch.crawler.fs.
↪test.unit.tika.TikaDocParserTest -Dtests.method="testExtractFromRtf" -Dtests.locale=ga-
↪IE -Dtests.timezone=Canada/Saskatchewan
```

You can just run the test again using the same seed to make sure you always run the test in the same context as before.

24.3.8 Tests options

Some options are available from the command line when running the tests:

- `tests.leaveTemporary` leaves temporary files after tests. `false` by default.
- `tests.parallelism` how many JVM to launch in parallel for tests. `auto` by default which means that it depends on the number of processors you have. It can be set to `max` if you want to use all the available processors, or a given value like 1 to use that exact number of JVMs.
- `tests.output` what should be displayed to the console while running tests. By default it is set to `onError` but can be set to `always`
- `tests.verbose` `false` by default
- `tests.seed` if you need to reproduce a specific failure using the exact same random seed
- `tests.timeoutSuite` how long a single can run. It's set by default to `600000` which means 5 minutes.
- `tests.locale` by default it's set to `random` but you can force the locale to use.
- `tests.timezone` by default it's set to `random` but you can force the timezone to use, like `CEST` or `-0200`.

For example:

```
mvn install -rf :fscrawler-it \
  -Dtests.output=always \
  -Dtests.locale=fr-FR \
  -Dtests.timezone=CEST \
  -Dtests.verbose \
  -Dtests.leaveTemporary \
  -Dtests.seed=E776CE45185A6E7A
```

24.4 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
```

If you want to skip the check, you can run with `-Dossindex.fail=false`:

```
mvn clean install -Dossindex.fail=false
```

If a CVE needs a temporary exclusion, you can add it to the `excludeVulnerabilityIds` list of the `ossindex` maven plugin in the `pom.xml` file:

```
<configuration>
    <excludeVulnerabilityIds>
        <!-- LINK TO CVE and COMMENT -->
        <excludeVulnerabilityId>CVE-2022-1471</excludeVulnerabilityId>
    </excludeVulnerabilityIds>
</configuration>
```

24.5 Docker build

The docker images build is ran when calling the maven `package` phase. If you want to skip the build of the images, you can manually use the `docker.skip` option:

```
mvn package -Ddocker.skip
```

24.6 DockerHub publication

To publish the latest build to DockerHub you can manually call `docker:push` maven task and provide credentials `docker.push.username` and `docker.push.password`:

```
mvn -f distribution/pom.xml docker:push \
    -Ddocker.push.username=yourdockerhubaccount \
    -Ddocker.push.password=yourverysecuredpassword
```

Otherwise, if you call the maven `deploy` phase, it will be done automatically. Note that it will still require that you provide the credentials `docker.push.username` and `docker.push.password`:

```
mvn deploy \
    -Ddocker.push.username=yourdockerhubaccount \
    -Ddocker.push.password=yourverysecuredpassword
```

You can also provide the settings as environment variables:

- `env.DOCKER_USERNAME` or `DOCKER_USERNAME`
- `env.DOCKER_PASSWORD` or `DOCKER_PASSWORD`

CHAPTER TWENTYFIVE

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 Python3 installed.

Assuming you have Python3 already, install Sphinx:

```
$ pip install -r docs/requirements.txt
```

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.

To update the requirements file if you changed the `requirements.in` file, run:

```
$ cd docs
$ pip-compile requirements.in
```

CHAPTER

TWENTYSIX

RELEASE THE PROJECT

To release the project, run:

```
$ release.sh
```

The release script will:

- Create a release branch
- Replace SNAPSHOT version by the final version number
- Commit the change
- Run tests against all supported elasticsearch series
- Build the final artifacts using release profile (signing artifacts and generating all needed files)
- Tag the version
- Prepare the announcement email
- Deploy to https://s01.oss.sonatype.org/
- Prepare the next SNAPSHOT version
- Commit the change
- Release the Sonatype staging repository
- Merge the release branch to the branch we started from
- Push the changes to origin
- Announce the version on https://discuss.elastic.co/c/annoucements/community-ecosystem

You will be guided through all the steps.

You can add some maven options while executing the release script such as `-DskipTests` if you want to skip the tests while building the release.

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

Only developers with write rights to the DockerHub repository can push the Docker images.

CHAPTER TWENTYSEVEN

RELEASE NOTES

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.

CHAPTER

TWENTYEIGHT

VERSION 2.10

28.1 New

- You can now remove a document in Elasticsearch using FSCrawler `_document` endpoint. Thanks to dadoonet.
- Implement our own HTTP Client for Elasticsearch. Thanks to dadoonet.
- Add option to set path to custom tika config file. Thanks to iadcode.
- Support for Index Templates. Thanks to dadoonet.
- Support for Aliases. You can now index in an alias. Thanks to dadoonet.
- Support for Access Token and Api Keys instead of Basic Authentication. Thanks to dadoonet.
- Allow loading external jars. This adds a new `external` directory from where jars can be loaded to the FSCrawler JVM. For example, you could provide your own Custom Tika Parser code. Thanks to dadoonet.
- Add temporal informations in folder index. Thanks to bdauvissat

28.2 Fix

- `fs.ocr.enabled` was always false. Thanks to ywjung.
- Do not hide YAML parsing errors. Thanks to dadoonet.

28.3 Deprecated

- The `_upload` REST endpoint has been deprecated. Please now use the `_document` endpoint. Thanks to dadoonet.
- Support for Elasticsearch 6.x is deprecated. Thanks to dadoonet.
- Support for Basic Authentication is deprecated. Thanks to dadoonet.

28.4 Updated

- Add full support for Elasticsearch Elasticsearch 8.13.2, Elasticsearch 7.17.19, Elasticsearch 6.8.23. Thanks to dadoonet.
- Update to Tika Tika 2.9.2. Thanks to dadoonet.

28.5 Removed

- Remove the specific distributions depending on Elastic version. Thanks to dadoonet.

Thanks to `@dadoonet`, `@ywjung`, `@iadcode`, `@bdauvissat` for this release!

CHAPTER

TWENTYNINE

VERSION 2.9

29.1 New features

- Add more default displayed fields in Workplace Search. Thanks to dadoonet.

29.2 Documentation

- Improve documentation for settings. Thanks to cbb-colab.

29.3 Changes

- Switch to the new sonatype service. Thanks to dadoonet.
- Bump log4j to 2.17.1. Thanks to dadoonet.
- Update to Tika 2.2.1. Thanks to dadoonet.
- Update to Elasticsearch 7.16.2. Thanks to dadoonet.

Thanks to `@cbb-colab`, `@dadoonet` for this release!

CHAPTER THIRTY

VERSION 2.8

30.1 New features

- Update ocr.rst, the path was wrong and not working. Thanks to sahin52.
- Add section Workaround for huge temporary files. Thanks to dfbm.

30.2 Fixed Bugs

- Fix starting fscrawler with Docker. Thanks to dadoonet.
- fix: not working optional libraries (e.g. jpeg2000). Thanks to NickUfer.
- Add procps apt package to container install. Thanks to cwperry.
- File logs missing in docker container. Thanks to helsonxiao.

30.3 Changes

- Bump log4j-core from 2.14.1 to 2.15.0.
- Update to Tika 2.1. Thanks to dadoonet.

Thanks to `@sahin52`, `@dfbm`, `@NickUfer`, `@cwperry`, `@helsonxiao`, `@dadoonet` for this release!

CHAPTER

THIRTYONE

VERSION 2.7

A lot of works happened for this release. More than 800 commits since version 2.6.

Note: FSCrawler can now send documents to Workplace Search, meaning that users can benefit from a powerful and centralized interface to search for local documents in addition to enterprise documents like Dropbox, Google Drive…

This version is mainly meant to work with Elasticsearch 7.x but you might be able to use it with 6.8 version.

The mapping for folders have changed and is more aligned with the mapping for documents.

Docker images are now generated from the build.

- FSCrawler comes now with an elasticsearch 7.x implementation.
- FSCrawler supports Workplace Search 7.x.
- FSCrawler also supports YAML format for jobs (default).
- The elasticsearch 6.x implementation does not support elasticsearch versions prior to 6.7. If you are using an older version, it's better to upgrade or you need to "hack" the distribution and replace all elasticsearch/lucene jars to the 6.6 version.
- FSCrawler does not follow symbolic links anymore. You need to set explicitly `fs.follow_symlink` to `true` if you wish revert to the previous behavior.
- The mapping for elasticsearch 6.x can not contain anymore the type name.
- We removed the Elasticsearch V5 compatibility as it's not maintained anymore by elastic.
- You need to use a recent JVM to run FSCrawler (Java 11 as a minimum. Java 15+ recommended)
- The mapping for the folders changed and is now consistent with the mapping for documents. If you are already using FSCrawler, you will need to first remove the existing `*_folder` indices and remove or edit the default settings files in `~/_default/7/_settings_folder.json` and `~/_default/6/_settings_folder.json` or any job specific setting file like `~/.fscrawler/{job_name}/_mappings/7/_settings_folder.json` or `~/.fscrawler/{job_name}/_mappings/6/_settings_folder.json`.

Thanks to `@CircuitGuy`, `@Einsteinder`, `@JLLeitschuh`, `@Maijin`, `@TommyLike`, `@aram535`, `@chrissound`, `@dadoonet`, `@gaiadas`, `@helsonxiao`, `@ian-cameron`, `@isaac-ipl`, `@janhoy`, `@jetersen`, `@k3ninho`, `@kikkauz`, `@mario-89`, `@muraken720`, `@shahariaazam`, `@toto1310`, `@wrathagom`, `Aram Mirzadeh`, `Erwan Arzur` and `fco-at-801217851326` for this release!

CHAPTER

THIRTYTWO

VERSION 2.6

- FSCrawler comes now with multiple distributions, depending on the elasticsearch cluster you're targeting to run.
- `elasticsearch.nodes` settings using `host`, `port` or `scheme` have been replaced by an easier notation using `url` setting like `http://127.0.0.1:9200`. You will need to modify your existing settings and use the new notation if warned.

CHAPTER

THIRTYTHREE

VERSION 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 THIRTYFOUR

VERSION 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.

CHAPTER
THIRTYFIVE

VERSION 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"
  }
}
# 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.

CHAPTER

THIRTYSIX

VERSION 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`.

CHAPTER

THIRTYSEVEN

LICENSE

CHAPTER THIRTYEIGHT

INCOMPATIBLE 3RD PARTY LIBRARY LICENSES

To support JPEG 2000 (JPX/JP2) images, you need to manually add jai-imageio-jpeg2000:1.4.0 library to the `external` directory:

```
cd external
wget https://repo1.maven.org/maven2/com/github/jai-imageio/jai-imageio-jpeg2000/1.4.0/
↪jai-imageio-jpeg2000-1.4.0.jar
```

See pdfbox documentation for more details about the license details.

CHAPTER THIRTYNINE

SPECIAL THANKS

Thanks to JetBrains for the IntelliJ IDEA License!