



# Open Source License Compliance by Open Source Software

## FOSSology SCA Integration

*Presenters:*

*Gaurav Mishra <gmishx@gmishx.in>*

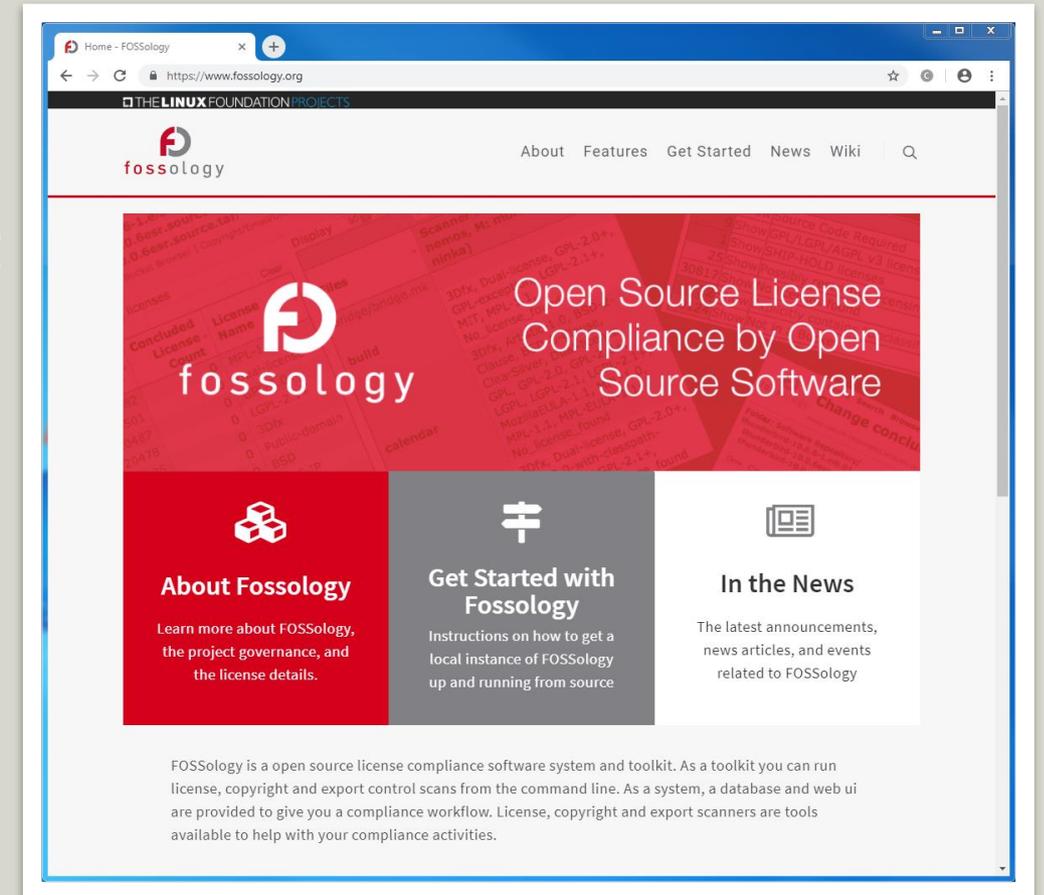
*Shaheem Azmal M MD <shaheem.azmal@gmail.com>*

*Anupam Ghosh <anupamghosh.ind@gmail.com>*

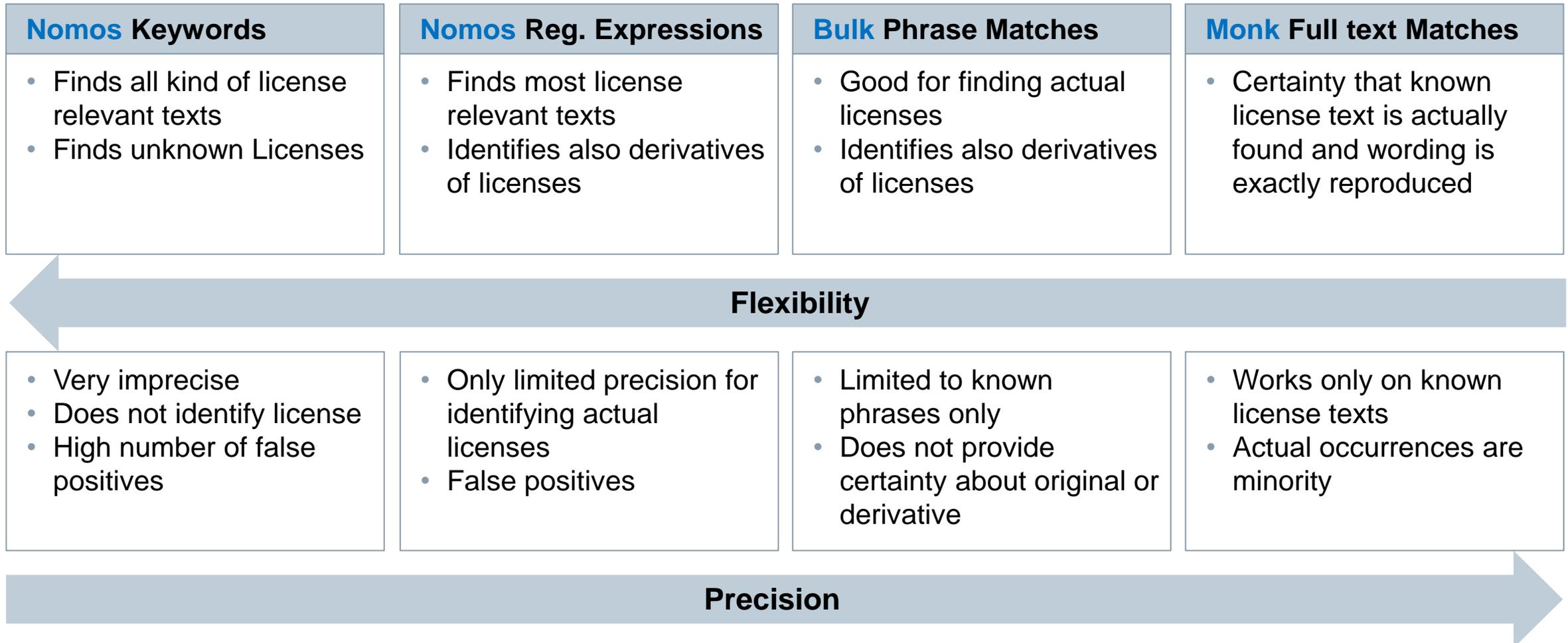
# FOSSology – Linux Foundation Collaboration Project

[www.fossology.org](https://www.fossology.org)

- 2008 initial publication by HP
- 2015 Linux Foundation Collaboration Project
- It is a Linux Application
- Different tasks for OSS license compliance
  - Scanning for licenses
  - Copyright, authorship, e-mails
  - ECC statements
  - Generation of documentation
  - Export and import SPDX files



# Features: Two License Scanners: Nomos and Monk



# Ojo, a scanner to detect SPDX-License-IDs

```
#include<stdio.h>
/*
 * Written by John Doe
 * SPDX-License-Identifier: Apache-2.0
 */
int main() {
    printf("Hello World\n");
    return 0;
}
```

**Gets detected as Apache-2.0 – reuse.software**

# Combine Ojo results with the other scanners

## Automatize

- The Ojo information can be combined with the other findings
- If no other scanner found a contradicting statement, the result can be concluded

- MIME-type Analysis (Determine mimetype of every file. Not needed for license)
- Monk License Analysis, scanning for licenses performing a text comparison
- Nomos License Analysis, scanning for licenses using regular expressions
- Ojo License Analysis, scanning for licenses using SPDX-License-Identifier
- Package Analysis (Parse package headers)

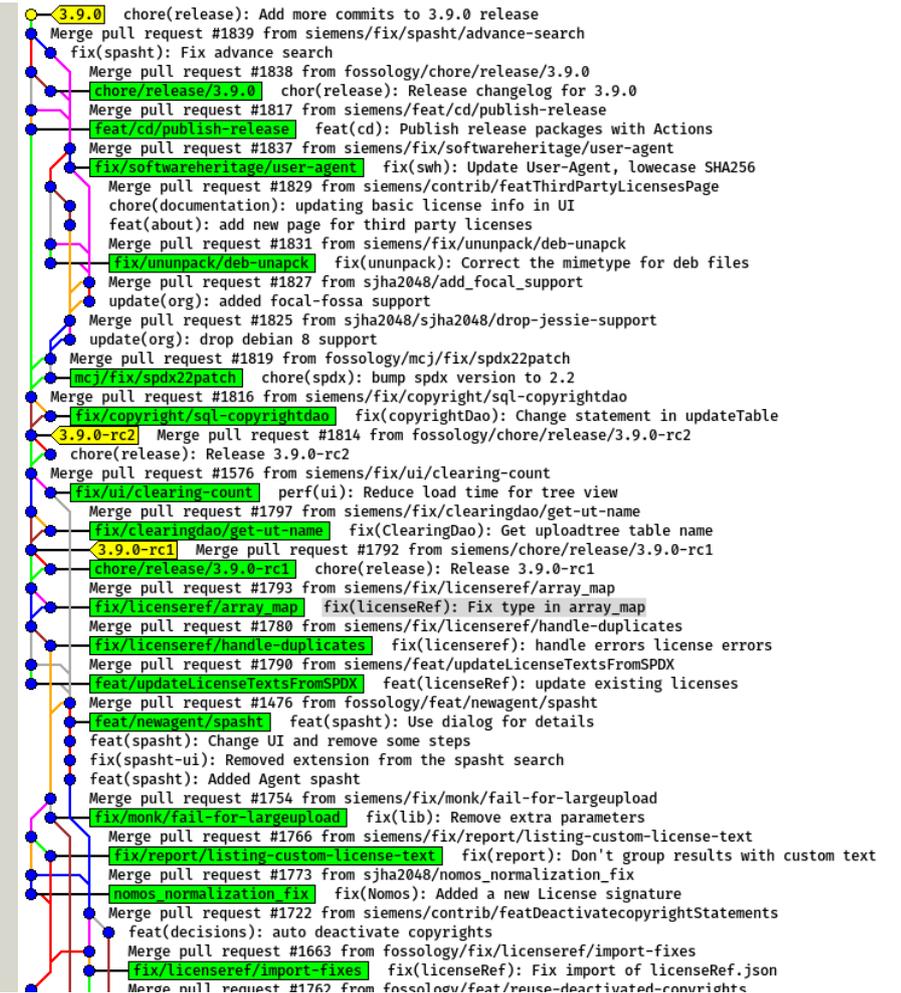
### 7. Automatic Concluded License Decider ⓘ, based on

- ... scanners matches if all Nomos findings are within the Monk findings
- ... scanners matches if Ojo findings are no contradiction with other findings
- ... bulk phrases from reused packages
- ... new scanner results, i.e. decisions were marked as work in progress if no

# FOSSology Updates – What is new in 3.9.0

<https://github.com/fossology/fossology/releases>

- New agent Spasht, works with ClearlyDefined.io
- PostgreSQL 12 support
- Ability to specify GIT branch in Upload from VCS
- Reuse of deactivated copyrights
- Remove OpenSSL dependency, use libgcrypt
- Support for Ubuntu Focal Fossa (20.04)
- Obligations now refer to license conclusions
- Auto deactivation of copyrights for irrelevant files
- Display time in browser's timezone
- Ability to export Copyright CSV



# FOSSology – Of course you can automate!

## REST API

- Manage folders, uploads
- Trigger scans and options
- Download reports
- More info at:
  - <https://www.fossology.org/get-started/basic-rest-api-calls/>
  - <https://github.com/fossology/fossology/wiki/FOSSology-REST-API>
- (complete flow explained)

## REST Clients

- Available in many languages
  - Python (fossology-python)
  - C# (FOSSology.REST.dotnet)
  - Shell (FOSSology.REST.shell)
- FOSSdriver
  - Not only what REST API can do
    - ... but also manage bulk scans
- More info at:
  - <https://github.com/fossology>
- Write your own Python workflow

## Command line tools

- Many functions and agents have command line interfaces
  - Nomos an Monk license scanners
  - Copyright scanner
  - License listings
  - ...
- Upload and download tools



## ***Uploading a package***

# Upload using REST Clients

## Shell Client

```
File Edit View Bookmarks Settings Help
All options:
-c , --description ) Upload description
-d , --debug       ) Debug mode
-e , --extra-debug ) Extra Debug mode
-f , --folder      ) Folder in which the upload will be added
-g , --group-name  ) Fossology group name
-h , --help        ) This help
-i , --input       ) Filename to upload
-k , --insecure    ) Skip certificate check in curl command
-n , --username    ) Fossology username
-p , --password    ) Fossology password
-r , --rest-url    ) Full address to Rest API service
-R , --reuse       ) Enable reuse
-s , --site-url    ) Fossology portal address
                    ) Enables printing the resulting Fossology URL
-t , --api-token   ) API Access Token
-u , --git-url     ) Url GIT repository address
-v , --version     ) Print current version
drax@empress:~/development/rest-shell (master)$ cat upload-all-from-folder.sh
FILES_TO_UPLOAD=$(ls /path/to/zips)
for file in $FILES_TO_UPLOAD
do
./upload-rest.sh --api-token 'my.secret.token' --rest-url https://fossology.tld/repo/api
/v1 --folder 3 --group-name fossy --input $file
done
drax@empress:~/development/rest-shell (master)$
```

## Python Client

```
1 # Import FOSSology object
2 from fossology import Fossology
3 # Import Access levels
4 from fossology.obj import AccessLevel
5
6 # Create request object
7 foss = Fossology(FOSS_URL, FOSS_TOKEN, username)
8 # Upload all files inside the folder
9 uploads = list()
10 for root, dirs, files in os.walk(folder_to_scan, followlinks=True):
11     for file in files:
12         file_to_upload = os.path.join(root, file)
13         my_upload = foss.upload_file(
14             foss.rootFolder,
15             file=file_to_upload,
16             description="Dependency of my project",
17             access_level=AccessLevel.PUBLIC,
18             wait_time=30
19         )
20         uploads.append(my_upload)
```

# Upload using CLI Tool

## cp2foss

```
File Edit View Bookmarks Settings Help
using):
-S          = upload from subversion repo
-G          = upload from git repo
--user string = user name
--pass string = password

FOSSology storage options:
-f path    = folder path for placing files (e.g., -f 'Fedora/ISOs/Disk 1')
           You do not need to specify your top level folder.
           All paths are under your top level folder.
-A         = alphabet folders; organize uploads into folder a-c, d-f, etc.
-AA num    = specify the number of letters per folder (default: 3); implies -A
-n name    = (optional) name for the upload (default: name it after the file)
-d desc    = (optional) description for the update

FOSSology processing queue options:
-Q         = list all available processing agents
-q         = specify a comma-separated list of agents, or 'all'
NOTE: By default, no analysis agents are queued up.
-T         = TEST. No database or repository updates are performed.
           Test mode enables verbose mode.
-I         = ignore scm data scanning

drax@empres:~/development/fossology (master)$ sudo /usr/local/bin/cp2foss \
> --username fossy --password fossy -f 'Components/MyProject/2.1.3' \
> -d 'Dependency for My Project v2.1.3' '/path/to/zips/*.zip'
```

## Upload sources

- All the methods supports upload from
  - From file
  - From URL
  - From GIT
  - From SVN
  - From Server



## ***Scanning a package***

# Scanning upload using REST Clients

## Shell Client

```
File Edit View Bookmarks Settings Help
drax@empres:~/development/rest-shell (master)$ cat json-templates/scan-options.json
{
  "analysis": {
    "bucket": true,
    "copyright_email_author": true,
    "ecc": true,
    "keyword": true,
    "mime": true,
    "monk": true,
    "nomos": true,
    "package": true
  },
  "decider": {
    "nomos_monk": true,
    "bulk_reused": true,
    "new_scanner": true
  },
}

drax@empres:~/development/rest-shell (master)$ ./upload-rest.sh \
> --api-token 'my.secret.token' --rest-url https://fossology.tld/repo/api/v1 \
> --folder 3 --group-name fossy --input 'file-to-scan.tar.gz'
```

## Python Client

```
1 # Import FOSSology object
2 from fossology import Fossology
3 # Import Access levels
4 from fossology.obj import AccessLevel
5
6 # Create request object
7 foss = Fossology(FOSS_URL, FOSS_TOKEN, username)
8 # Create scan specification
9 job_spec = {
10     "analysis": {
11         "copyright_email_author": True,
12         "monk": True,
13         "nomos": True,
14         "ojo": True
15     }
16 }
17 # Set the folder where upload is
18 upload_folder = foss.detail_folder(upload_to_scan.folderid)
19 # Trigger the scan
20 foss.schedule_jobs(upload_folder, upload_to_scan, job_spec, group='fossy',
21 wait=True, timeout=60)
```

# Scanning upload using CLI Tool

## fossjobs

```
File Edit View Bookmarks Settings Help
drax@empress:~/development/fossology (master)$ sudo /usr/local/bin/fossjobs -h
fossjobs [options]
Options:
-h          :: help, this message
-v          :: verbose output
-a          :: list available agent tasks
-A string  :: specify agent to schedule (default is everything from -a)
             The string can be a comma-separated list of agent tasks.
-u          :: list available upload ids
-U upload  :: the upload identifier for scheduling agent tasks
             The string can be a comma-separated list of upload ids.
             Or, use 'ALL' to specify all upload ids.
-D upload  :: the upload identifier for scheduling delete tasks
             The string can either be 'ALL', a string (the upload_pk),
             or an array of upload_pk's if multiple -D's were specified.
--username string :: user name
--password string :: password
--groupname string :: group name
-c string  :: Specify the directory for the system configuration
drax@empress:~/development/fossology (master)$ sudo /usr/local/bin/fossjobs \
> --username fossy --password fossy -A 'agent_monk,agent_nomos,agent_ojo,agent_copyright'\
> -U 204 --groupname 'fossy' █
```

## Scanning options

- Analysis
  - Various agents
- Decider
  - Automatic conclusions
  - nomos-monk, ojo, new scanner
  - Bulk reuse
- Reuse
  - Copy clearings from existing upload



## ***Analysing the results***

# Analysing results using cURL

## Upload summary

```
File Edit View Bookmarks Settings Help
drax@empres:~/development/fossology (master)$ curl --silent --request GET \
> "http://localhost/repo/api/v1/uploads/195/summary" \
> --header "groupName: fossy" \
> --header "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2MTEyNTM3OTksIm5iZiI6MTYwODU3NTQwMCwianRpIjoiTURrdU13PT0iLCJzY29wZSI6IndyaXRlIn0.MjkoGbdhAZkgxrdsK20kIl1CTmTkMsOf1WurGn7RQKo" | python -m json.tool
{
  "id": 195,
  "uploadName": "virtualenv-20.0.31.tar.gz",
  "mainLicense": "MIT",
  "uniqueLicenses": 14,
  "totalLicenses": 246,
  "uniqueConcludedLicenses": 9,
  "totalConcludedLicenses": 356,
  "filesToBeCleared": 0,
  "filesCleared": 474,
  "clearingStatus": "Open",
  "copyrightCount": 106
}
drax@empres:~/development/fossology (master)$
```

## Upload licenses

```
File Edit View Bookmarks Settings Help
drax@empres:~/development/fossology (master)$ curl --silent --request GET \
> "http://localhost/repo/api/v1/uploads/195/licenses?agent=nomos,monk,ojo&containers=false" \
> --header "groupName: fossy" \
> --header "Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJleHAiOjE2MTEyNTM3OTksIm5iZiI6MTYwODU3NTQwMCwianRpIjoiTURrdU13PT0iLCJzY29wZSI6IndyaXRlIn0.MjkoGbdhAZkgxrdsK20kIl1CTmTkMsOf1WurGn7RQKo" | python -m json.tool
[
  {
    "filePath": "virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/src/virtualenv/seed/wheels/embed/pip-19.1.1-py2.py3-none-any.whl/pip/_vendor/chardet/escprober.py",
    "findings": {
      "scanner": [
        "LGPL-2.1+"
      ],
      "conclusion": [
        "LGPL-2.1+"
      ]
    }
  },
  {
    "filePath": "virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/src/virtualenv/seed/wheels/embed/pip-19.1.1-py2.py3-none-any.whl/pip/_vendor/chardet/enums.py",
    "findings": {
      "scanner": [

```

# Analysing results using REST Client

## Upload summary

```
1 # Import FOSSology object
2 from fossology import Fossology
3 # Import Access levels
4 from fossology.obj import AccessLevel
5
6 # Create request object
7 foss = Fossology(FOSS_URL, FOSS_TOKEN, username)
8
9 # Get the upload object
10 upload = foss.detail_upload(195, group='fossy')
11 # Get the upload summary
12 upload_summary = foss.upload_summary(upload, group='fossy')
13
14 # Print the upload details
15 print(f"Upload name: {upload_summary.uploadName}")
16 print(f"Upload main license: {upload_summary.mainLicense}")
17 print(f"Upload state: {upload_summary.clearingStatus}")
18 print(f"Unique licenses in upload: {upload_summary.uniqueLicenses}")
19 print(f"Clearing: {upload_summary.filesCleared}" +
20       | f"/{upload_summary.filesToBeCleared}")
21
```

## Upload licenses

```
1 # Import FOSSology object
2 from fossology import Fossology
3 # Import Access levels
4 from fossology.obj import AccessLevel
5
6 # Create request object
7 foss = Fossology(FOSS_URL, FOSS_TOKEN, username)
8
9 # Get the upload object
10 upload = foss.detail_upload(195, group='fossy')
11 # Get the upload licenses
12 upload_licenses = foss.upload_licenses(upload, group='fossy',
13                                       agent="nomos,monk,ojo", containers=False)
14
15 # Print the licenses
16 for license in upload_licenses:
17     print(f"In file {license.filepath} =>")
18     print("  Scanner: ", end='')
19     print(license.findings.scanner)
20     print("  Conclusion: ", end='')
21     print(license.findings.conclusion)
```

# Analysing results using CLI tools

## Nomos Licenses

```
File Edit View Bookmarks Settings Help
drax@empres:~/development/fossology (master)$ sudo /usr/local/bin/fo_nomos_license_list
--username fossy --password fossy -u 195
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/tox.ini: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/setup.py: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/setup.cfg: MIT
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/.readthedocs.yml: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/README.md: MIT
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/pyproject.toml: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/.pre-commit-config.yaml: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/PKG-INFO: MIT
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/LICENSE: MIT
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/.dockerignore: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/.coveragerc: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/codecov.yaml: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/tests/conftest.py: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/tests/unit/test_util.py: No_license_found
virtualenv-20.0.31.tar.gz/virtualenv-20.0.31.tar/virtualenv-20.0.31/tests/unit/test_run.py: No_license_found
```

## Other tools

- fo\_nomos\_license\_list
  - All findings by nomos scanner
  - Can exclude file paths
- fo\_monk\_license\_list
  - All findings by monk scanner
  - Can exclude file paths
- fo\_copyright\_list
  - All copyright findings
  - Can filter copyright content



## ***Generating reports***

# Generating reports using REST Clients

## Shell Client

```
File Edit View Bookmarks Settings Help
<program> <authentication> <rest api url> <upload id> <report format> [other options...]
<program> -v

Authentication methods:
- Token      : -t <...>
- User + Password : -n <...> -p <...>

Rest API URL: -r <rest_api_url>
                Ex. https://service-fqdn/repo/api/v1

All options:
-d, --debug           ) Debug mode
-e, --extra-debug    ) Extra Debug mode
-F, --report-format  ) Format of the report to download
-h, --help           ) This help
-k, --insecure       ) Skip certificate check in curl command
-n, --username       ) Fossology username
-o, --output-dir     ) Output directory to download report
-p, --password       ) Fossology password
-r, --rest-url       ) Full address to Rest API service
-t, --api-token      ) API Access Token
-u, --upload-id      ) Upload ID on the server
-v, --version        ) Print current version
drax@empres:~/development/rest-shell (master)$ ./download-rest.sh \
> --api-token 'my.secret.token' --rest-url https://fossology.tld/repo/api/v1 \
> --upload-id 195 --report-format 'unifiedreport' --output-dir reports
```

## Python Client

```
1 # Import FOSSology object
2 from fossology import Fossology
3 # Import report formats
4 from fossology.obj import ReportFormat
5
6 # Create request object
7 foss = Fossology(FOSS_URL, FOSS_TOKEN, username)
8
9 # Get the upload object
10 upload = foss.detail_upload(195, group='fossy')
11
12 # Generate the report
13 report_id = foss.generate_report(upload, ReportFormat.UNIFIEDREPORT,
14                                 group='fossy')
15
16 # Download the report
17 report_content, report_name = foss.download_report(report_id)
18 with open(report_name, "w+") as report_file:
19     report_file.write(report_content)
20
21
```

Report formats available: dep5, sdx2, sdx2tv, readmeoss, unifiedreport

# There is more at [github.com/fossology](https://github.com/fossology)

## Atarashi

- Standalone license scanner
- Written in Python
- Implement multiple text statistics and information retrieval algorithms
- More info at: <https://github.com/fossology/atarashi>

## FOSSologySlides

- Slides for Presenting FOSSology
- Make your presentation with FOSSology
- Material for a 1-day training
- More info at: <https://github.com/fossology/FOSSologySlides>

## Nirjas

- Python library and tool
- Extract source code and comments
- Supports 25 languages
- Differentiate single line, multiline and continued single line comments
- More info at: <https://github.com/fossology/Nirjas>

# Questions? – Consider to “Star Us”!



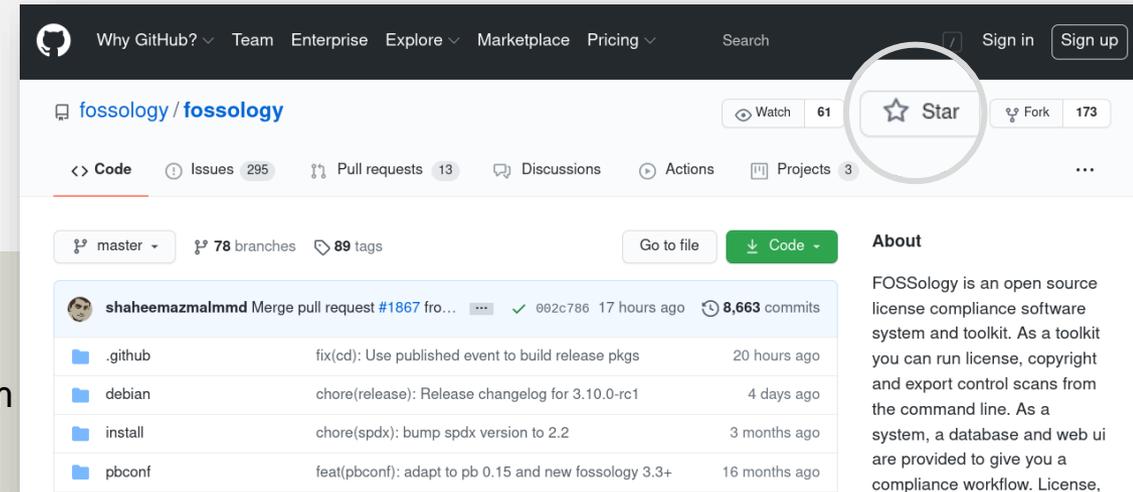
Gaurav Mishra  
Siemens AG  
gmishx@gmishx.in



Shaheem Azmal  
Siemens AG  
shaheem.azmal@gmail.com



Anupam Ghosh  
Siemens AG  
anupamghosh.ind@gmail.com



FOSSology links

<https://www.fossology.org/>

<https://github.com/fossology/fossology>

FOSSology - YouTube

<https://www.youtube.com/channel/UCZGPJnQZVnEPQWxOuNamLpw>