User documentation

Welcome to the user documentation of AMOS firmware scraper. This page is meant to teach you how to use the application.

For learning how to build the application, please refer to the build documentation.

Configure the application

In the `src` folder you will find the configuration file `config.json`. This file is meant to define user input data.

General parameters

`config.json` supports the following general parameters:

```
"database": {
  "user": "user",
  "password": "password"
},
"download_dir": "./download_dir"
"max_products": 5
"log_level": "DEBUG"
```

Instead of providing your MySQL credentials in the configuration file, you may alternatively use the environment variables `MYSQL_USER` and `MYSQL_PASSWORD`. These environment variables take precedence over the values defined in the configuration file.

If `download_dir` is not set, firmware images will be downloaded into `./downloads` of the project's root.

`max_products` limits the number of firmware products each vendor will scrape until aborting the scraping operation. This is meant for testing purposes as scraping all firmware of a vendor can be very time consuming.

`log_level` defines the logging level used for console output. You may choose between these values:

```
["DEBUG", "INFO", "IMPORTANT", "WARNING", "ERROR", "CRITICAL"]
```

Vendor-specific parameters

The configuration file also contains a list of all vendors that are supposed be scheduled for scraping. Per default, `config.json` lists all vendors currently supported.

These are the parameters of a single vendor:

```
"vendors": [
  {
      "name": "ABB",
      "class_name": "ABBScraper",
      "active": true,
      "interval": "0",
      "last_update": "2023-02-01",    # (yyyy-mm-dd)
      "next_update": "2023-02-01",    # (yyyy-mm-dd)
      "max_products": 10              # use null to inherit default
  },
  #...
```

For users, the parameters `active`, `interval` and `max_products` are of special interest. `active` provides a switch for deactivating vendors without removing their parameters from `config.json`. `interval` defines in how many days a vendor will be rescheduled for scraping after a successful run. When set, `max_products` overwrites the default defined by the general parameter.

`name` and `class_name` are predefined and should normally not be changed. `last_update` and `next_update` are automatically updated by the application.

Run the application

You can run the application natively by navigating into the root directory of the project and running

```
python -m src.core
```

Alternatively, you can run the application with Docker by navigating into the root directory of the project and running

```
docker-compose up --build
```

The application will initialize the database, load general parameters and the vendors to be scheduled from the configuration file `config.json` and start the scraping. You can see the incoming scraped firmware in the logs.

```
❯ python3 -m src.core
2023-02-07 23:30:04,428 -      core.py:228 -      INFO - Scheduled scrapers: ['SchneiderElectricScraper', 'SwisscomScraper']
2023-02-07 23:30:04,476 -      core.py: 44 -      INFO - Initialized core and DB.
2023-02-07 23:30:04,477 -      core.py:235 - IMPORTANT - Next: SchneiderElectricScraper
2023-02-07 23:30:06,902 - schneider.py:229 - IMPORTANT - Started scraping
2023-02-07 23:30:08,949 - schneider.py:237 -      INFO - Successfully accessed entry point URL https://www.se.com/us/en/download/doc-group-type/3587139-Software%20&%20Firmware/?doc
2023-02-07 23:30:10,430 - schneider.py:166 -      INFO - Successfully scraped firmware EGX300 v4.400 Firmware and 4.450 Disk Update https://www.se.com/us/en/download/document/EGX30
2023-02-07 23:30:10,810 - schneider.py:184 -      INFO - Successfully scraped firmware M241 M251 Firmware https://www.se.com/us/en/download/document/M241_M251_Firmware_v4.0.6.39/
2023-02-07 23:30:11,353 - schneider.py:166 -      INFO - Successfully scraped firmware Rack Air Removal Unit SX Firmware v3.1.1 https://www.se.com/us/en/download/document/DBUE-764Q
2023-02-07 23:30:11,768 - schneider.py:166 -      INFO - Successfully scraped firmware Dry Contact I/O SmartSlot Card (AP9613) Firmware v2.1.0 https://www.se.com/us/en/download/doc
2023-02-07 23:30:12,585 - schneider.py:166 -      INFO - Successfully scraped firmware Network Management Card BMSHVA3 v352 Executable https://www.se.com/us/en/download/document/DB
2023-02-07 23:30:12,907 - schneider.py:166 -      INFO - Successfully scraped firmware InRow RC 300mm v621 Firmware Executable https://www.se.com/us/en/download/document/MFOI-AZMQ2
2023-02-07 23:30:13,244 - schneider.py:166 -      INFO - Successfully scraped firmware KVM Firmware v1.0.089 for KVM2123P and KVM2116P https://www.se.com/us/en/download/document/MF
2023-02-07 23:30:13,752 - schneider.py:184 -      INFO - Successfully scraped firmware Network Management Card v6.5.0 complete firmware executable for Symmetra 3- Phase UPS with AP
2023-02-07 23:30:14,178 - schneider.py:166 -      INFO - Successfully scraped firmware Rack Air Removal Unit SX Firmware v3.7.1 https://www.se.com/us/en/download/document/MFOI-AJRV
2023-02-07 23:30:14,543 - schneider.py:166 -      INFO - Successfully scraped firmware Edge Box HMI Firmware https://www.se.com/us/en/download/document/EdgeBoxHMI_Firmware_V1.1.0/
```

Depending on the amount of firmware provided by the vendors and the performance of their corresponding websites, this can take many hours.

After scraping, the download function will iterate over all vendors of this run and, based on the scraped download link, download firmware images into the directory given in `config.json`.

```
2023-02-07 23:30:30,768 -      core.py:269 - IMPORTANT - Start firmware download.
2023-02-07 23:30:30,768 -      core.py:272 - IMPORTANT - Download directory: /[redacted]repo/downloads
2023-02-07 23:30:31,878 -      core.py:151 - IMPORTANT - Next: SchneiderElectric
2023-02-07 23:30:33,846 -      core.py:206 -      INFO - [1/17] Successfully downloaded 11_v4.450.zip
2023-02-07 23:30:37,014 -      core.py:206 -      INFO - [2/17] Successfully downloaded 12_Image_SD_Card_M241_V4.0.6.39.7z
2023-02-07 23:30:41,468 -      core.py:206 -      INFO - [3/17] Successfully downloaded 13_Image_SD_Card_M251_V4.0.6.39.7z
2023-02-07 23:30:50,458 -      core.py:206 -      INFO - [4/17] Successfully downloaded 14_M241_4.3.9.11_18.08.10.01.seco
2023-02-07 23:30:53,967 -      core.py:206 -      INFO - [5/17] Successfully downloaded 15_M251_4.3.9.11_18.08.10.01.seco
```

Downloading firmware from all vendors requires sufficient storage (over 200 GB free storage recommended).

Check results

Results are written into the table `products` of the MySQL database `firmware`.

For example, try

```
mysql> SELECT count(*), manufacturer FROM products GROUP BY manufacturer;
|+----------+--------------+
|| count(*) | manufacturer |
|+----------+--------------+
||      301 | foscam       |
|+----------+--------------+
```

Re-running the application

By re-running the application, all vendors corresponding to the schedule will be executed and start scraping sequentially. Only new products will be stored in the database and downloaded afterwards. Since this open-source projects relies heavily on the vendors' mood, please do not re-run it in a relentless manner.

Known issues

- Vendor's website is not available or not performant enough, leading to timeouts (scraping can be rescheduled)
- Vendor's website may change over time, causing the specific vendor module to be outdated
- Vendors can identify the scraping attempt and block your IP or apply other technical measures to prevent scraping