KenSentMe

Overview

KenSentMe is a graphical interface for The Sleuth Kit, based on an existing graphical interface called Autopsy.

It differs from the existing software by allowing quick configuration for collaboration, and the ability to acess content on any browser.

It is being developed by Pedro Ferreira, a Cybersecurity Student, as a final project for his master's degree, while doing an internship at VOID Software.

Dependencies

- The Sleuth Kit
- photorec - usually installed with testdisk package
- ImageMagick
- FFMPEG
- Apache Solr
- Pure-FTPd
- Git.

Every other dependency is handled through maven repositories.

Setup

The Sleuth Kit

The Sleuth Kit packages are available for some linux distributions but we only support version 4.8.0, so it should be compiled from this fork.

Before compiling TSK, the following packages should also be installed (and compiled from source):

- libuna
- libewf
- afflib
- libvmdk
- libvhdi.

libvmdk's repo includes a zlib folder; it's not recommended to compile it.

Run `autoreconf -f -i` before `./configure` for libvmdk and libvhdi.

Make sure `ant` is installed before compiling TSK, as it is required to generate the java bindings.

Run `ldconfig` to sync libraries after installing TSK.

Jars

Inside the Quarkus repository, cd into `jars` directory and run the following commands:

```
mvn install:install-file -Dfile=sleuthkit-4.8.0.jar -DgroupId=org.sleuthkit -DartifactId=sleuthkit -Dversion=4.8.0 -Dpackaging=jar
mvn install:install-file -Dfile=Rejistry-1.0-SNAPSHOT.jar -DgroupId=com.williballenthin.rejistry -DartifactId=Rejistry -Dversion=1.0-SNAPSHOT -Dpackaging=jar
mvn install:install-file -Dfile=sevenzipjbinding.jar -DgroupId=net.sf.sevenzipjbinding -DartifactId=sevenzipjbinding -Dversion=4.65-1.06-rc-extr-only -Dpackaging=jar
mvn install:install-file -Dfile=sevenzipjbinding-Linux-amd64.jar -DgroupId=net.sf.sevenzipjbinding -DartifactId=sevenzipjbinding-Linux-amd64 -Dversion=4.65-1.06-rc-extr-only -Dpackaging=jar
```

Quarkus

- Clone the repository
- Edit `config.properties` and set the desired properties
- Compile JAR with `./mvnw clean compile quarkus:build`
- Copy `config.properties` to `target`
- Run with `java -Djava.awt.headless=true -jar kensentme-1.0.0-runner.jar`.

React

- Clone the repository
- Edit `src/config.js` file with the server IP
- Run `npm run build`
- Copy `build` to the location used by your web server.

Solr

- Extract `solr.7z` inside `/var/solr` directory

Pure-FTPd

- Make sure MySQL support is enabled and is using the same database as Quarkus

License

KenSentMe is under the Apache License, Version 2.

Documentation

org.sleuthkit.autopsy package

The contents of this package were adapted from the existing Autopsy platform and follow a very similar directory structure.

Not all original classes were transfered into this version and almost all user interface elements were removed except for module configuration panels.

These classes may contain altered functions, which means updating to newer versions of Autopsy must be done in a function by function manner, or at least while checking carefully for differences in each class, to make sure the platform's functionalities remain unchanged.

The code present in this package is taken from the 4.13.0 release of Autopsy.

Contents

- casemodule - Case class, case actions such as data source addition and tag interactions
- centralrepository - Central database used to compare hashes of files between cases
- coreutils - Classes used to handle data
- ingest - Procedures and entities that make data ingestion possible
- keywordsearch - Procedures and entities required to interact with Solr indexing server
- modules - Pre-included ingestion modules
- python - Jython Module Loader, allows the functioning of python modules, contains extra code for module management
- report - Procedures entities and modules used for generating reports.

pt.voidsoftware.kensentme package

The contents of this package are responsible for handling api requests.

Contents

- authentication - Login, registration, 2FA, recovery and activation
- casemodule - Main functionalities of the platform, contains five resources responsible for providing endpoints for general case actions, explorer actions, board actions, content actions and csv generation; each resource is accompanied by a utility class responsible for gathering the requested information or running the required procedure
- file - Simple persisted entity that can be provided as a parameter for ingest/report modules
- ftp - Class required by Pure-FTPd to grant access to the FTP server, associated in one to one to users, also contains the user's ftp directory

- settings - Add-on modules management and tag management
- team - Users can be added to teams and teams can be assigned to cases
- user - User, ftp and 2fa management.

react code

Contains all the UI elements, and interacts with the java api.

Contents

actions

Redux requests for auth, case information, snackbar notification, and users/teams/cases.

assets

Images and styles. Styles are composed by a main file which addresses most of the components, a modal specific file and a context menu specific file.

components

`screens` contain the different pages available in the web app; names are self descriptive.

`elements` contains components that are reused, or were created to make the parent module easier to read.

Directories inside elements:

- esm - Material tree view components that were modified to achieve the desired behavior
- main - Main components of the project, board contains the table/thumbnail viewer, content contains the content viewers, explorer contains the case explorer, and the toolbar contains most of the user interactions as well as a socket handler
- modals - Modals used throughout the whole project.

reducers

Redux storage for auth, case information, snackbar notification, and users/teams/cases.