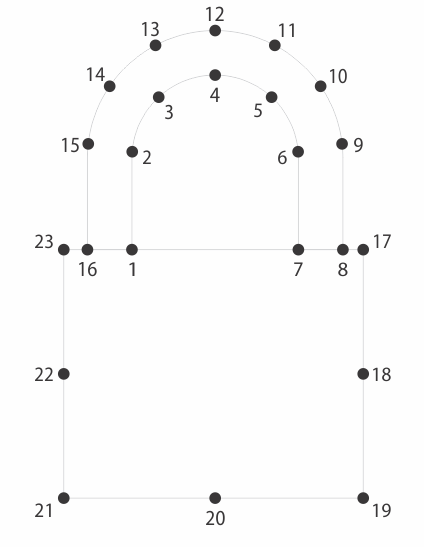
Marie Skłodowska-Curie Research and

Innovation Staff Exchange Equality and Citizenship



PROTASIS

Restoring Trust in the Cyber Space:

A Systems Security Proposal

**D2.5: CyberSecurity Repository[[1]](#footnote-2)†**

**Abstract**: This report summarizes the effort put by the PROTASIS project in the *Cybersecurity Repository*: a platform to allow simplified access to research on cybersecurity.

|  |  |
| --- | --- |
| Contractual Date of Upload | October 2017 |
| Actual Date of Upload | October 2017 |
| Deliverable Security Class | Public |
| Editor | Davide Quarta |
| Contributors | All *PROTASIS* partners |
| Quality Assurance | Name2 Surname2 |

The *PROTASIS* consortium consists of:

|  |  |  |
| --- | --- | --- |
| FORTH | Coordinator | Greece |
| POLIMI | Beneficiary | Italy |
| VU | Beneficiary | The Netherlands |
| RUB | Beneficiary | Germany |
| TELEFONICA | Beneficiary | Spain |
| F-Secure | Beneficiary | Finland |
| MIT | TC Partner | USA |
| UCSB | TC Partner | USA |
| UIC | TC Partner | USA |
| Columbia University | TC Partner | USA |
| Northeastern University | TC Partner | USA |
| Stony Brook University | TC Partner | USA |
| Stevens | TC Partner | USA |
| Boston University | TC Partner | USA |

# Table of Contents

[Table of Contents 3](#__RefHeading___Toc401_1694561164)

[1Introduction 4](#__RefHeading___Toc403_1694561164)

[2Cybersecurity repository 5](#__RefHeading___Toc405_1694561164)

[2.1Authentication and browsing 5](#__RefHeading___Toc3030_932902391)

[2.2Projects 5](#__RefHeading___Toc409_1694561164)

[2.3Publications 6](#__RefHeading___Toc411_1694561164)

[2.4Files 9](#__RefHeading___Toc3032_932902391)

[2.5Data & Code 9](#__RefHeading___Toc413_1694561164)

[2.6Wiki 9](#__RefHeading___Toc417_1694561164)

[2.7User messages 11](#__RefHeading___Toc3034_932902391)

[3Technical details 13](#__RefHeading___Toc419_1694561164)

[3.1Deployment environment 13](#__RefHeading___Toc421_1694561164)

[3.2Authentication framework 13](#__RefHeading___Toc423_1694561164)

[3.2.1Views 13](#__RefHeading___Toc425_1694561164)

[4Feeding the repository 14](#__RefHeading___Toc435_1694561164)

[4.1Manually 14](#__RefHeading___Toc437_1694561164)

[4.2Automatically 14](#__RefHeading___Toc453_1694561164)

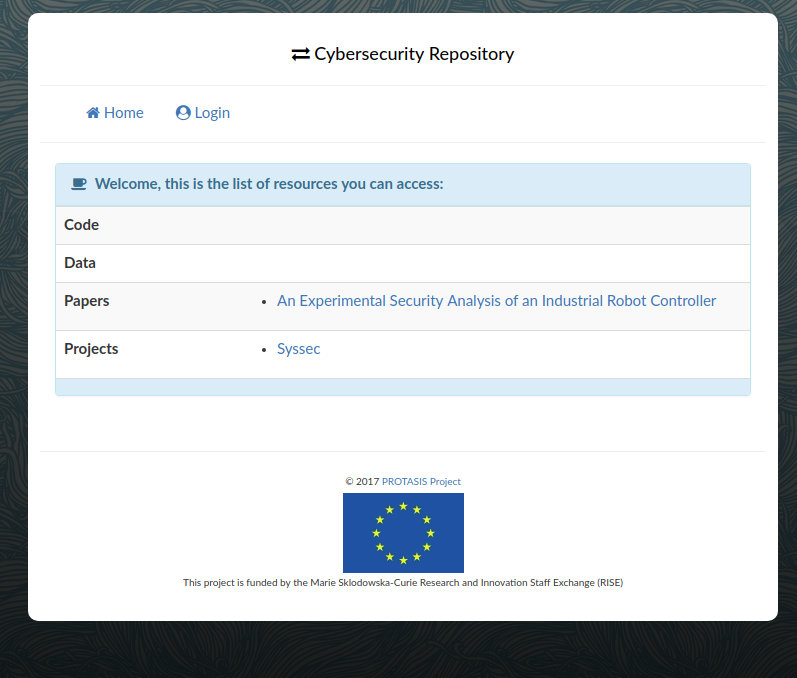
[5Limitations and future developments 18](#__RefHeading___Toc3036_932902391)

# Introduction

This report summarizes the dissemination, communication and exploitation activities carried out by the Protasis project in the May 2017–October 2011 period. Specifically, in the following pages we will describe the “*Cybersecurity Repository*” platform prototype.

This platform will allow keeping track of research outcomes and act as a receptacle for all the related informations.

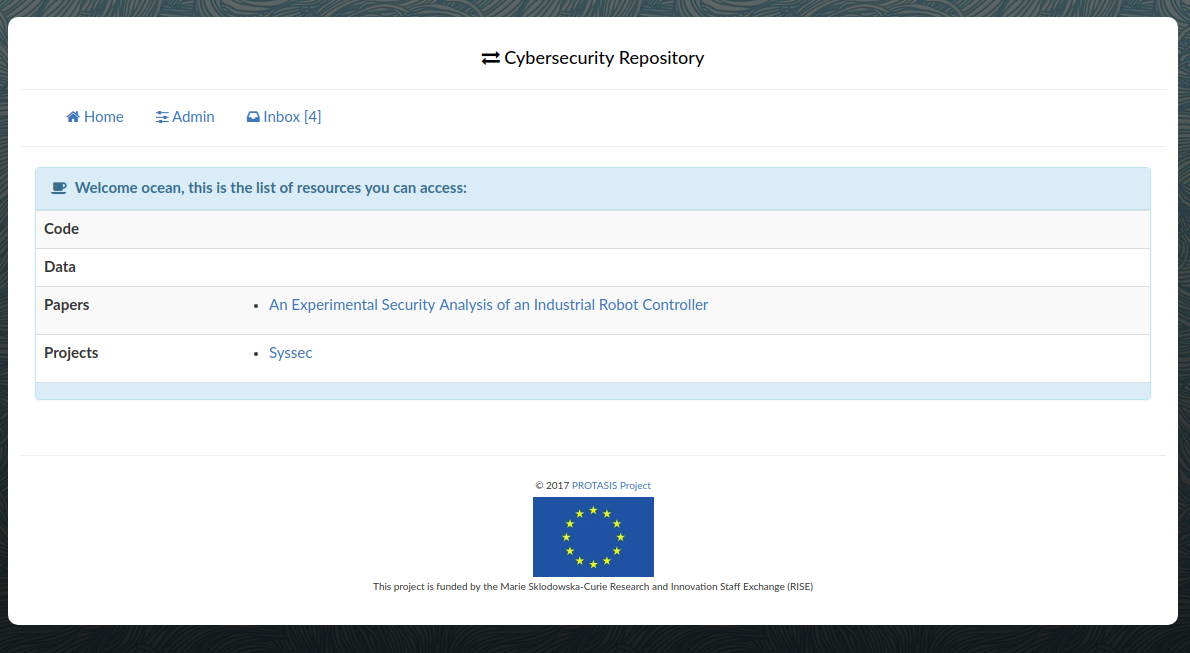
The source will be available at http://xxx.xx/protasis/cs\_repository/

  
Figure 1: the repository's home (not logged in)

# Cybersecurity repository

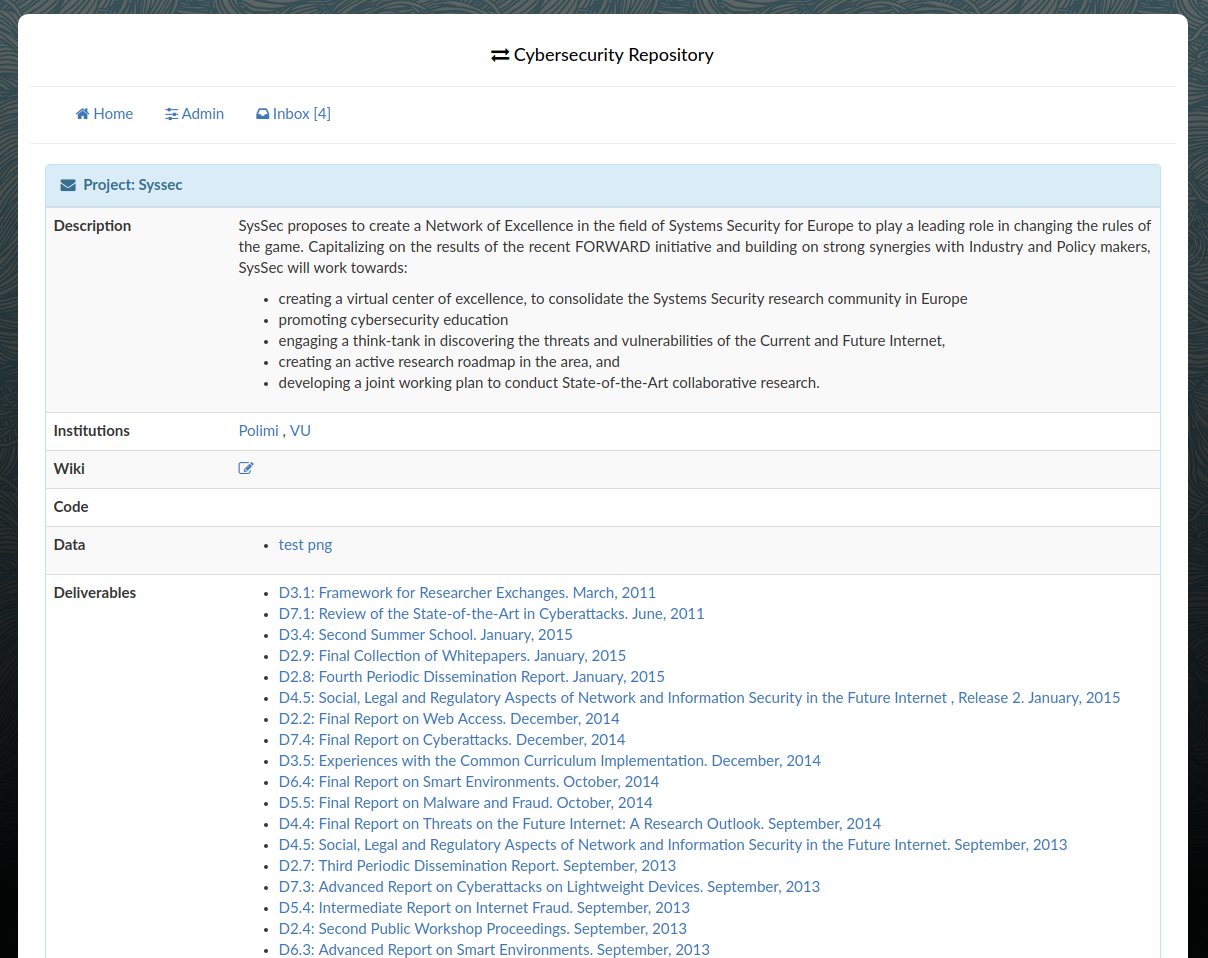
## Authentication and browsing

Once logged in the user will see all the resources he has access to, if the user is part of the staff group, he will be also able to modify the resources via the administration interface.

  
Figure 2 the home page of the repository when logged in

### Projects

A project is a “unit” to keep related material together. Papers, code and data can be added. Moreover a link to the internal *wiki* is generated to allow discussion and a more free space to keep material (i.e., teaching material).

  
Figure 3: The SysSec project in the repository

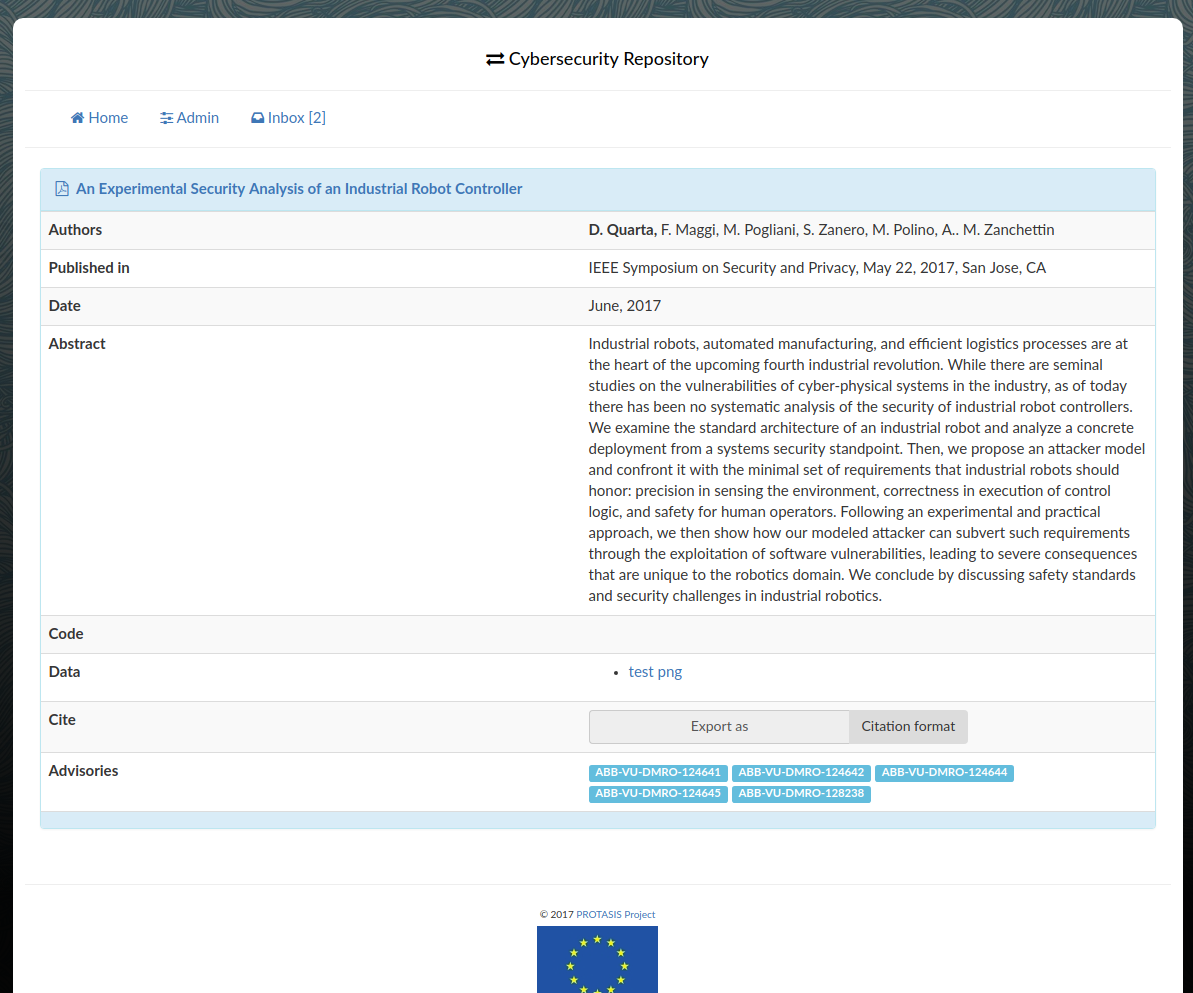
### Publications

The following items can be added as a pubblication:

* papers
* deliverables
* reports

More can easily be added by tweaking the code.

In Figure 4 a paper is show: the ID for vulnerability advisories can also be added to a publication where needed. The citations in bibtex format can be added to the project to allow easily exporting citations.

  
Figure 4: A paper stored in the repository

### Files

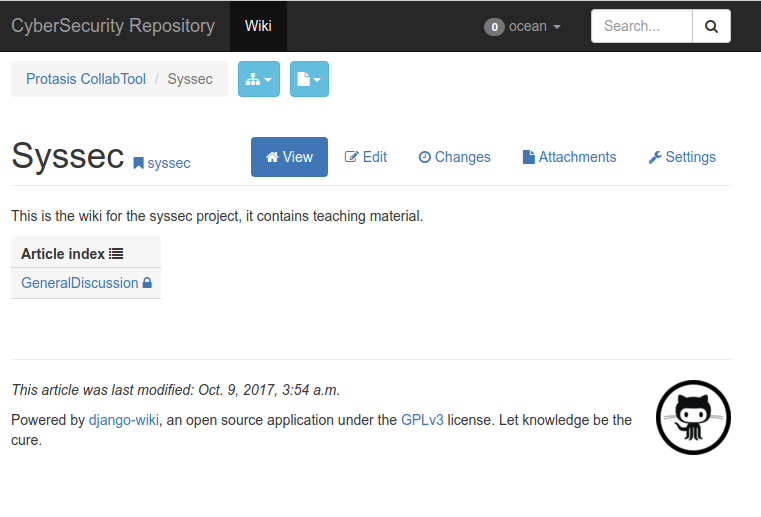
When an entry needs a particular file associated (i.e., the pdf for a publication), it can be stored in a file object, or an external url can be specified. This is handled transparently.

### Data & Code

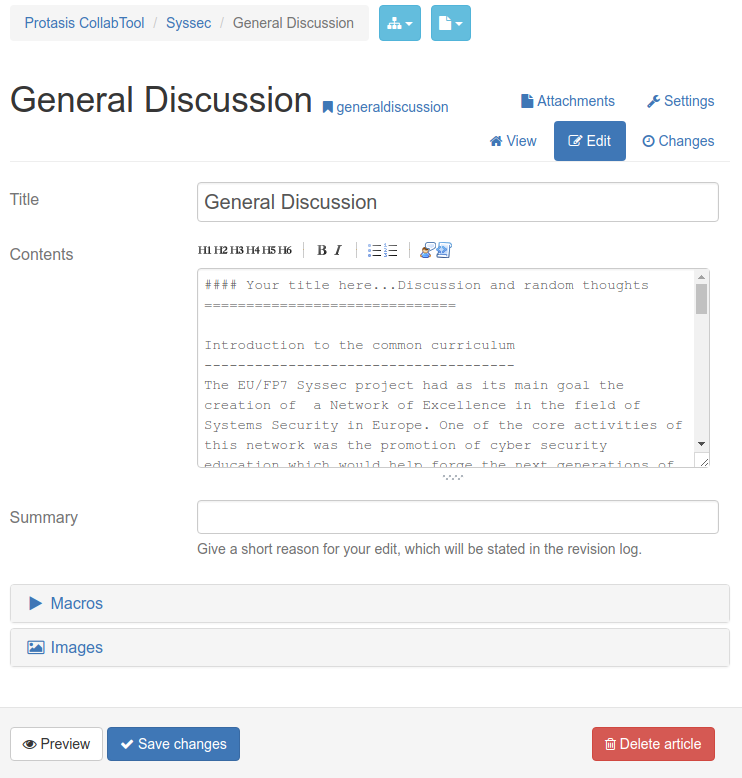
Publications and projects can also have several associated data and code packages. These are stored as multiple entries each one with their own set of access permissions to allow for fine grained access management.

## Wiki

Projects can also have a related wiki in which material and discussions can be dumped, the url for the related wiki is automatically generated for each entry, but the wiki entry will have to be manually generated. Tight integration with the wiki can be achieved by writing a plugin for the django-wiki software and can be planned for the next versions of the repostiory.

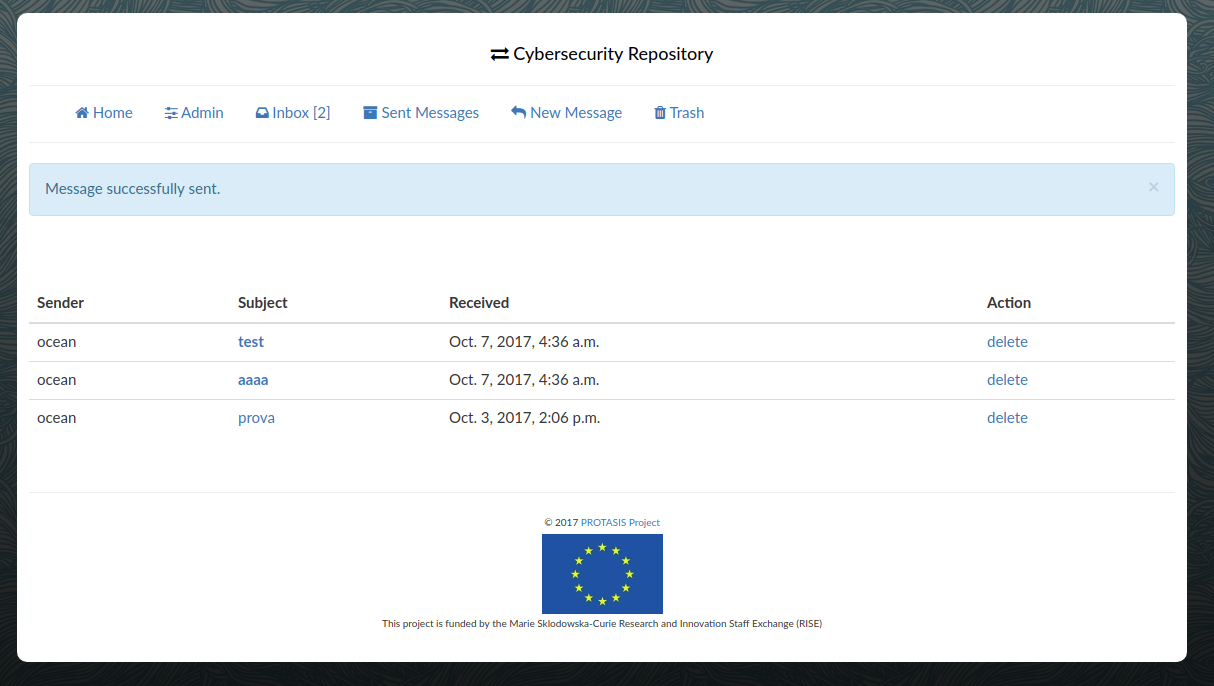
Illustration 1: The wiki for the SysSec project with an entry

Articles are written in markdown (<https://daringfireball.net/projects/markdown/>) a simple plaintext formatting syntax that will be automatically (and safely) converted to HTML.

  
Figure 5: editing the "General Discussion" entry in the SysSec wiki

## User messages

To foster communication and collaboration, the repository also provides a way for users of the platform to communicate via private messages.

  
Illustration 2: user messages interface

# Technical details

In this section we will introduce some technical details of the platform. Since the code of the project is written in a high-level language with a widely used web framework, the code is pretty self-explanatory. We will limit the description to critical design choices.

## Deployment environment

The repository is based on Python 2.7, Django 1.10 with several extensions, and the objects are stored using MySQL.

The repository has been developed using *virtual environments* and *pip*, a python modules repository (a requirements.txt is present in the git repository to ease deployment).

Moreover different *linters* (tools to ensure the “cleanliness” of the code) have been used during the development process, ensuring the code responds to the python community guidelines.

## Authentication framework

The out-of-the-box functionality offered by django is not flexible enough to handle fine grained permissions for objects. We developed a “framework” to allow easily adding new “*authenticable*” objects. Defining a authenticable models is done by adding the *AuthMixin* as a m*ixin* for the desired model.

All the *accessible* objects linked to a model are accessible using the “*get\_accessible*” class method, which will return all the accessible objects, and if the parameter “unlinked” is True it will return all objects that are *spurious* (i.e., they are not linked with any project/publication.

The “*accessible\_rel*” method will help getting all the *accessible* authenticable objects linked with the current model (i.e., publications, data, code…).

### Views

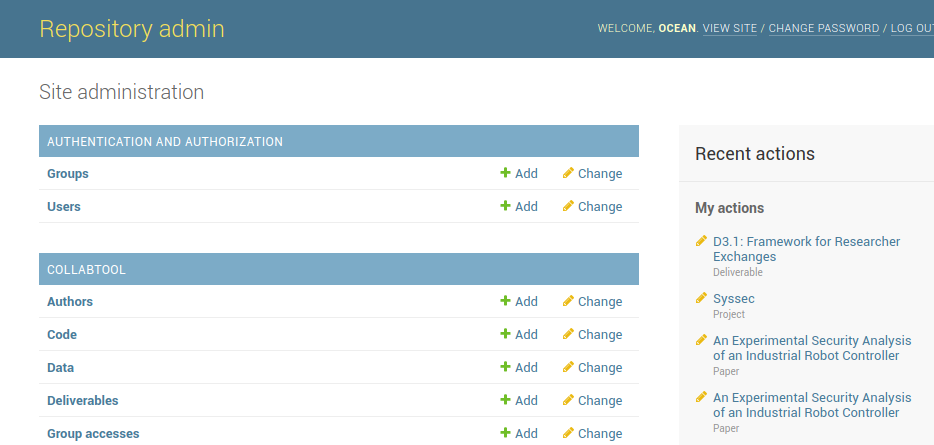
Access to the views is regulated via the “check\_group\_access” and “check\_data\_access” decorators, that will check for permission on the decorated views and will pass the object to the decorated view or raise a 404 if it is not possible to “fetch” the object.

# Feeding the repository

Feeding the repository can happen in two different ways, manually and automatically.

## Manually

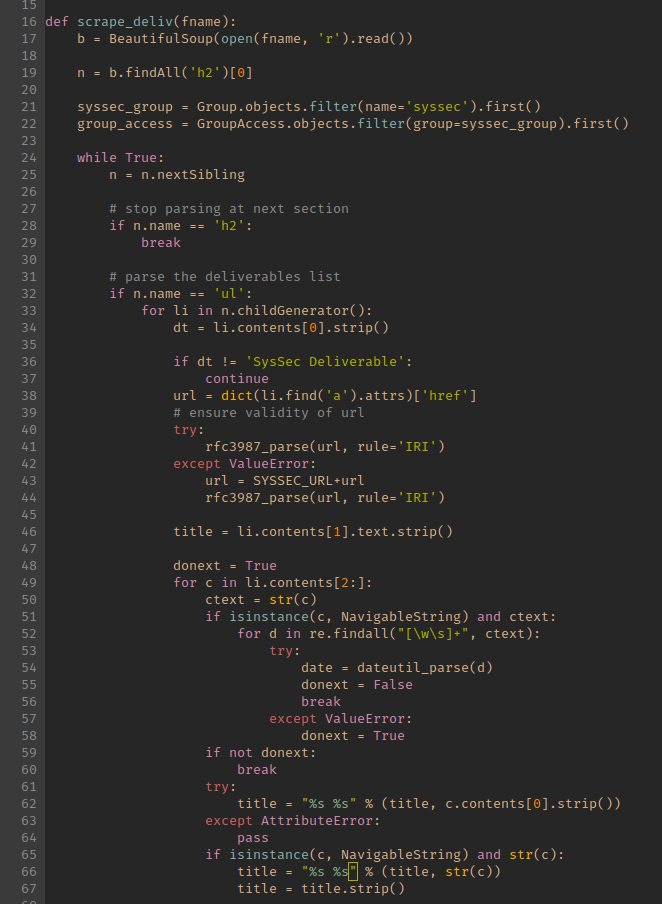
Any object can be added via the administration interface.

  
Figure 6: Repository administration interface

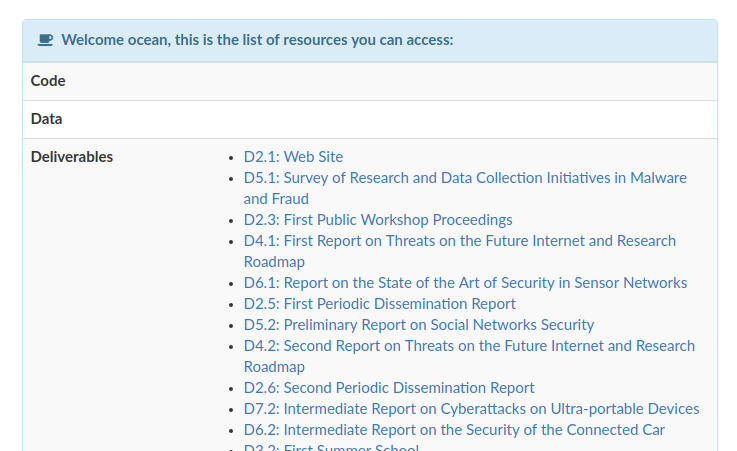
## Automatically

To automatically add objects, a custom django management command can be created, this is needed because of how django handles access to the application models.

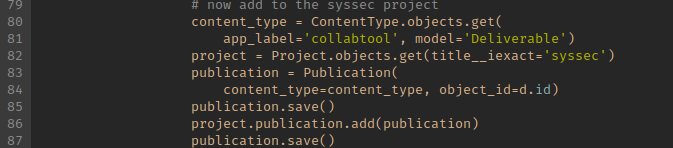
To programmatically create content we can for example scrape a web resource: to show this potentiality we developed an example that scrapes the deliverables of the SysSec project from its website publications page. The code implementing this example can be found inside the repository in the file: ***collabtool/management/commands/scrape\_deliverables.py****.*

  
Figure 7: scraping with beautifulsoupt and adding the deliverable

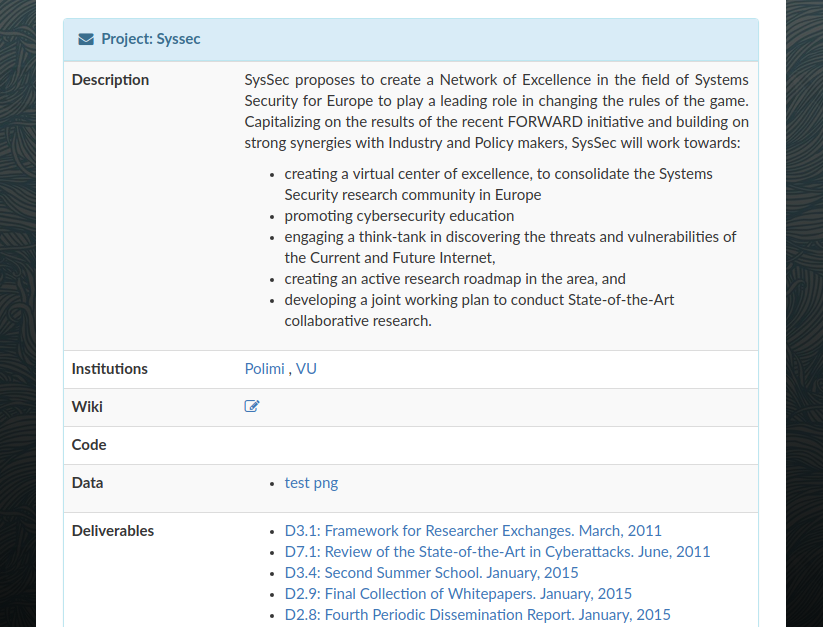
The scraped object will be saved and will be accessible to any user having read permissions for the syssec group.

  
Figure 8: the scraped deliverables not linked to any project

At this point the publications haven’t been added to the project so they will result as unsorted publications available to the user: we need to create then a Publication object (that is a generic container for publications), link it with the deliverable, and then add the Publication model to the project.

  
Figure 9: adding the publications to the project

Unfortunately this process cannot be simplified since the structure of the models depends on the capabilities offered by the Django ORM system. The final results can be seen in Figure Figure.

  
Figure 10: the SysSec project with the scraped deliverables

# Limitations and future developments

The repository is currently a beta version and some features are missing. The following are todos for which a basic infrastructure of the code/models already exists and won’t be hard to implement:

* add check on read for GroupAccess ''accessible'' operations (right now we only check that the user is part of the group, we will need to check the ‘read’ condition)
* handle write permissions to modify articles (check the ‘write’ condition)
* parse automatically PublicationBase model to export bibtex if the bibtex hasn't been inserted manually
* parse automatically bibtext (i.e. use bibtexparse python module) to automatically parse and insert a publication
* parse advisory ids and link to the advisory (i.e., CVE->[NVD...], MS->Microsoft, ABB→ABB)

Other desirable feature that could be added in the feature include:

* user comments
* statistics for projects/papers/authors
* better management system for permissions/insertions
* REST API
* feeds (xml…)

1. † This project is funded by the Marie Skłodowska-Curie Research and Innovation Staff Exchange (RISE). [↑](#footnote-ref-2)