D3.1 System and dependency analysis tool support adaptation

Grant Agreement number:	740723
Project acronym:	CS-AWARE
Project title:	A cybersecurity situational awareness and information sharing solution for local public administrations based on advanced big data analysis

Principal author:	Christian Wieser, UOULU, christian.wieser@oulu.fi
Co-author(s)	Toni Väisanen, UOULU, toni.vaisanen@oulu.fi

Document version:	1.0

Table of Contents

Executive Summary

This delivery describes improvements to GraphingWiki. GraphingWiki has been in use to describe complex systems and this document describes the adaptation for CS-AWARE. The adaptation consists of a new user interface. The implementation is continuing, as we still integrate feedback of the adaptation and extend the export functionality.

1 Introduction

In this document we describe the Dependency Mapper, a GUI application to manage graph data. The purpose of is to improve the usability of manual data collecting to enhance the existing GraphingWiki. GraphingWiki is an extension of the open-source wiki engine MoinMoin and was developed in 2006 to support the collecting and visualizing of data. It was successfully used in software protocol and malware analysis by introducing additional meta link syntax and (other capabilities).

This is achieved by providing a dynamic way of exploring and navigating data to replace the text-based wiki syntax approach. This is working also to bypass the graph generation on the server side. This improves the performance, since earlier studies have shown that the graph generation process had been the most time-consuming part on the server response and was therefore the major performance bottle neck.

Metadata is multi-valued, so you can define multiple values for one key. So, as an example, the metadata

```
likes:: ice cream

likes:: chocolate
```

the key "likes" will have values "ice cream" and "chocolate". This is then being further used to visualize graphs. The editing follows traditional wiki page editing, meaning raw text file editing using wiki syntax.

The majority of other software used in this area is used in the application/network monitoring segment. These products are focusing on automated monitoring of network traffic or agent-based monitoring of network/application nodes. While GraphingWiki has been used in context of socio-technical dependency discovery [R1] collecting interview data with and connecting this with technical details.

2 Implementation

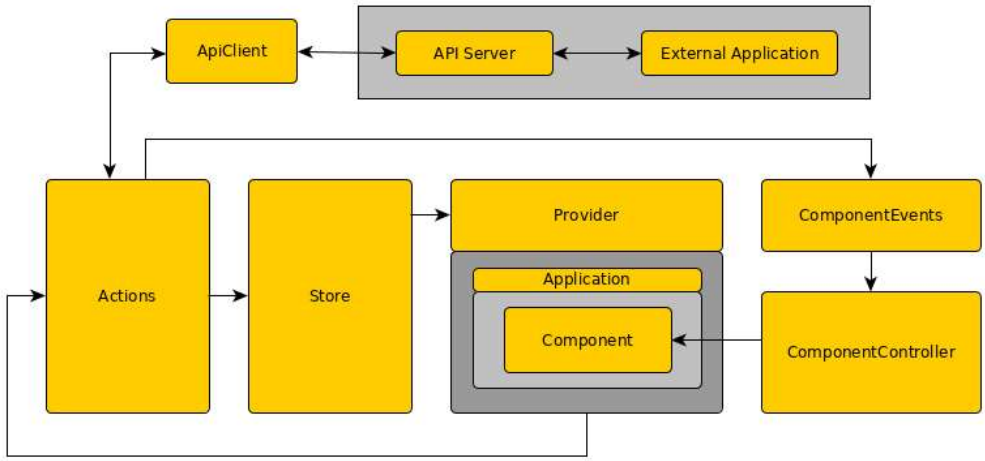
The Dependency Mapper is implemented as a stand-alone JavaScript application, because we wanted to reduce the explicit dependency to GraphingWiki. It is built with React JS, which is widely adopted front-end JavaScript library for building web user interfaces. The state management of the application is handled by Redux.

2.1 Externalities

The Dependency Mapper is unrelated to the server-side implementation and merely requires a HTTP API for data transfers.

2.2 Static structure of the developed software

The application follows the Redux pattern for state management, which has uni-directional data flow. The UI events trigger actions that are dispatched to the store ("state machine"), which in turn changes the application state. This state is distributed to the application through the redux provider component.



2.3 Used work

The following libraries allowed the swift progress in this development. When not mentioned otherwise, they were taken from the official node package manager website npmjs.com (accessed 11/10/2018) in its at that time current version.

2.3.1 axios

Promise based HTTP client for the browser and node.js. For handling network requests.

https://www.npmjs.com/package/axios

2.3.2 cytoscape

Cytoscape.js is a fully featured graph theory library. It contains a graph theory model and an optional renderer to display interactive graphs. This library was designed to make it as easy as possible for programmers and scientists to use graph theory in their apps, whether it's for server-side analysis in a Node.js app or for a rich user interface.

https://www.npmjs.com/package/cytoscape

2.3.3 cytoscape-cola

The Cola.js physics simulation layout for Cytoscape.js. The cola layout uses a force-directed physics imulation with several sophisticated constraints.

https://www.npmjs.com/package/cytoscape-cola

2.3.4 lodash

Lodash makes JavaScript easier by taking the hassle out of working with arrays, numbers, objects, strings, etc. Lodash's modular methods are great for:

- Iterating arrays, objects, & strings
- Manipulating & testing values
- Creating composite functions

https://www.npmjs.com/package/lodash

https://lodash.com/

2.3.5 react

React is a JavaScript library for creating user interfaces.

The `react` package contains only the functionality necessary to define React components. It is typically used together with a React renderer like `react-dom` for the web, or `react-native` for the native environments.

https://www.npmjs.com/package/react

2.3.6 react-dom

This package serves as the entry point to the DOM and server renderers for React. It is intended to be paired with the generic React package, which is shipped as `react` to npm.

https://www.npmjs.com/package/react-dom

2.3.7 react-markdown-renderer

Simple React component that renders Markdown, built with remarkable (a Markdown parser).

This is used to render markdown as HTML in the detail view description box.

https://www.npmjs.com/package/react-markdown-renderer

https://www.npmjs.com/package/remarkable

2.3.8 redux

Redux is a predictable state container for JavaScript apps. It helps writing applications that behave consistently, run in different environments (client, server, and native), and are easy to test. On top of that, it provides a great developer experience, such as live code editing combined with a time traveling debugger. It can be used Redux together with React, or with any other view library.
It is tiny (2kB, including dependencies).

https://www.npmjs.com/package/redux

2.3.9 react-redux

Official React bindings for Redux.
https://www.npmjs.com/package/react-redux

2.3.10 redux-thunk

Redux Thunk middleware allows you to write action creators that return a function instead of an action. The thunk can be used to delay the dispatch of an action, or to dispatch only if a certain condition is met. The inner function receives the store methods `dispatch` and `getState` as parameters.

https://www.npmjs.com/package/redux-thunk

2.3.11 styled-components

Utilising tagged template literals (a recent addition to JavaScript) and the power of CSS, `styled-components` allows you to write actual CSS code to style your components. It also removes the mapping between components and styles – using components as a low-level styling construct could not be easier!

https://www.npmjs.com/package/styled-components

2.4 Examples of the software in use

For an example, we used the dataset of the Larissa WP2 requirements analysis. Firstly, the graph was created using Graphingwiki and secondly, by using the Dependency Mapper. Table 1 has the pages listed with the related links. In this Table, blue text indicates a link to a wiki page that already exists and a Question mark ('?') in front of a camel case (WordStartsWithCapitalized) text is linking to a page which has not been created yet. The column "gwikicategory" refers to the Graphingwiki category links, which is a way to tag information.

Regional / International / Global -Section

	gwikicategory	Internet	Intranet	likes	object	Observed	OldLine	Server	VPN
ADSL Router	?CategoryGlobal	Internet							
End User MAN Switch	?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms, ?CategoryLarissa	SYZEFXIS Router	MAN VLAN10	KP4					
Endpoint VLAN	?CategoryGlobal, ?CategoryLarissa								
External dpt access HRMS and Genesis	?CategoryGenesis, ?CategoryGlobal, ?CategoryLarissa		KP Node						
Giannouli SYZEFXIS	?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms								VPN
Internet	?CategoryGlobal								
Koilada SYZEFXIS	?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms								VPN
KP Node	?CategoryGenesis, ?CategoryGlobal, ?CategoryLarissa		MAN VLAN10						
Main Switch	?CategoryFinanceFlow, ?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms, ?CategoryLarissa		End User MAN Switch			Router Gateway	SYZEFXIS Router		
MAN VLAN10	?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms, ?CategoryLarissa				KP1 (Culture Center), KP10 (Radio Station), KP12 (Green Spaces Dpt.), KP2 (L. Community), KP3 (Cementary), KP5 KP7 (Secondary Building), KP6 (Car Depot)				
Router Gateway	?CategoryFinanceFlow, ?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms, ?CategoryLarissa								
Server-2	?CategoryGlobal							Main Switch	
Server-3	?CategoryGlobal							Main Switch	
SYZEFXIS Router	?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms, ?CategoryLarissa	Internet							
VPN	?CategoryGenesis, ?CategoryGlobal, ?CategoryHrms								SYZEFXIS Router

Table 1: Larissa Regional / International / Global -section

The left most column of Table 1 translates to the active mapping "Global". Items in the column "gwikicategory" are translated to tags, which are assigned to each asset individually. The rest of the columns are assets that are tagged with other tags. Figure 1 shows a graph created based on Table 1, in Figure 2 it can be seen how the same data set translates to the new interface.

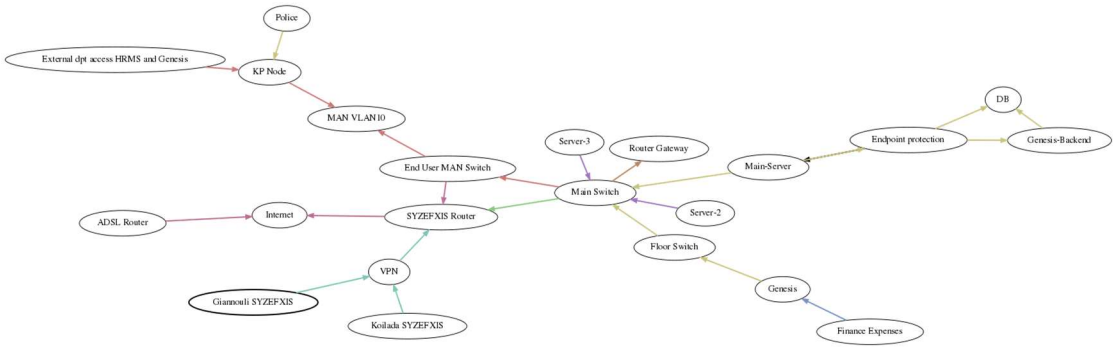


Figure 1: Larissa Regional / International / Global -section

The data presented in Figure 2 can be further explored by selecting items by clicking on the displayed lists. The list next to the graph shows the resources currently included in the active mapping, which are also drawn in the graph. Hovering the cursor above a list item will highlight the node with its neighbourhood (the nodes that the item hovered on is connected to) and a click will load the data to the detail view in the detail section of the view. In this view connections and tags related to the given detail is listed on the bottom right. These lists are further navigable, allowing the user to view the details of each of the items.

2.4.1 How mappings are created

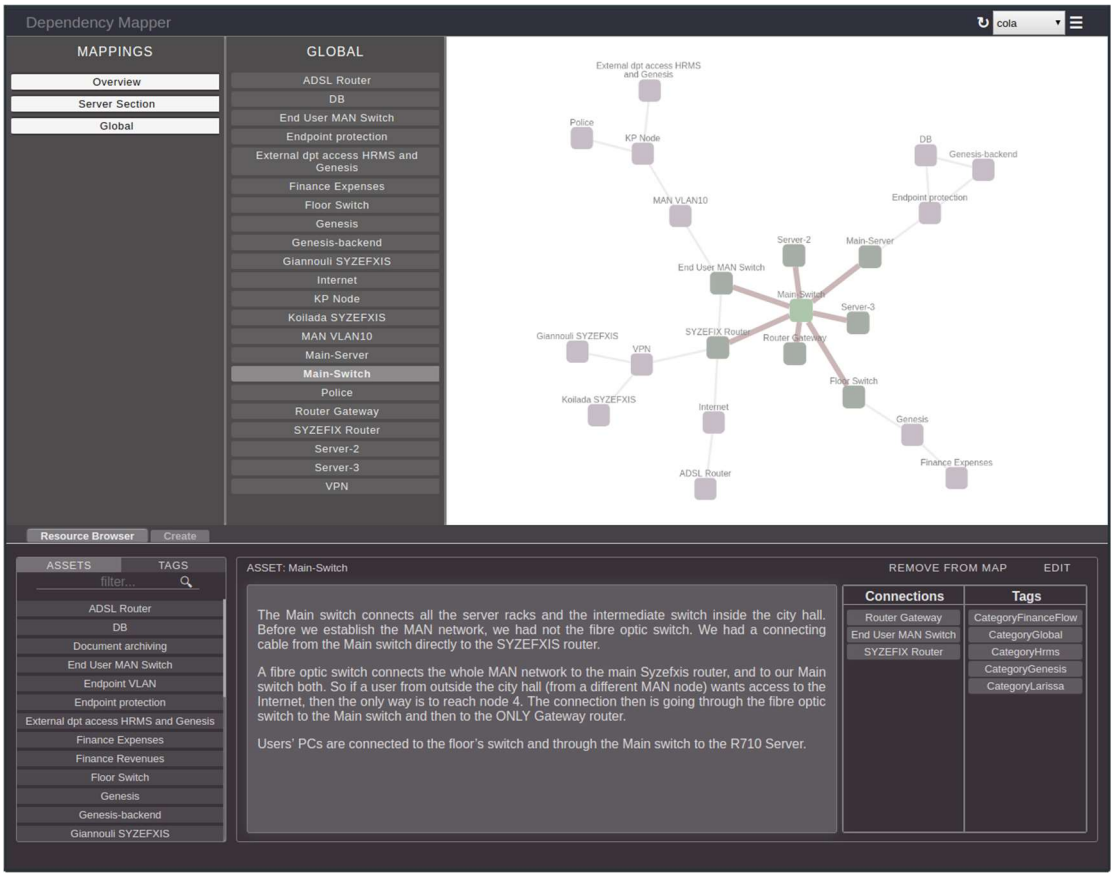


Figure 2: Larissa Regional / International / Global -section Translated in Dependency Mapper

In a further example, the "Server section" of the Larissa dataset was used. When a new mapping is created, the information of earlier used assets and tags are available for utilization. From the Table 2 it can be seen that the required assets to be created are: DB, Endpoint protection, Genesis-Backend, HRMS-Backend, and Main-Server.

Server-Section

«	Data «	gwikicategory «	object «
DB		?CategoryFinanceFlow, ?CategoryGenesis, ?CategoryHrms, ?CategoryLarissa, ?CategoryServer	
Endpoint protection	DB, Genesis-Backend, HRMS-Backend	?CategoryFinanceFlow, ?CategoryGenesis, ?CategoryHrms, ?CategoryLarissa, ?CategoryServer	
Genesis-Backend	DB	?CategoryFinanceFlow, ?CategoryGenesis, ?CategoryLarissa, ?CategoryServer	
HRMS-Backend	DB	?CategoryHrms, ?CategoryLarissa, ?CategoryServer	
Main-Server	Endpoint protection, Main Switch	?CategoryFinanceFlow, ?CategoryGenesis, ?CategoryHrms, ?CategoryLarissa, ?CategoryServer	ERP system, Oracle, Secondary Building, Synantec Endpoint Protection 10., Windows 2008 R2, no download, no mail, no upload

Table 2: Larissa Server-section

The first step in translating the table using the Dependency Mapper is to create a new mapping with the assets. The data and gwikicategory objects are attributes of the assets as connections and tags. Data column links are the connections of an asset and gwikicategory column links are represented by tags. In this case, four of five of the assets required by the mapping have already been created. These four assets can be included in the mapping by selecting them by clicking on the list items. The same mechanism applies to tags as well. The remaining asset: "Endpoint protection", had not been created yet. It must be created to be assigned to the mapping. This is done by writing the name of the asset in the text field saying "create new". By hitting the "Enter" key, the asset will be created and added to the selection.

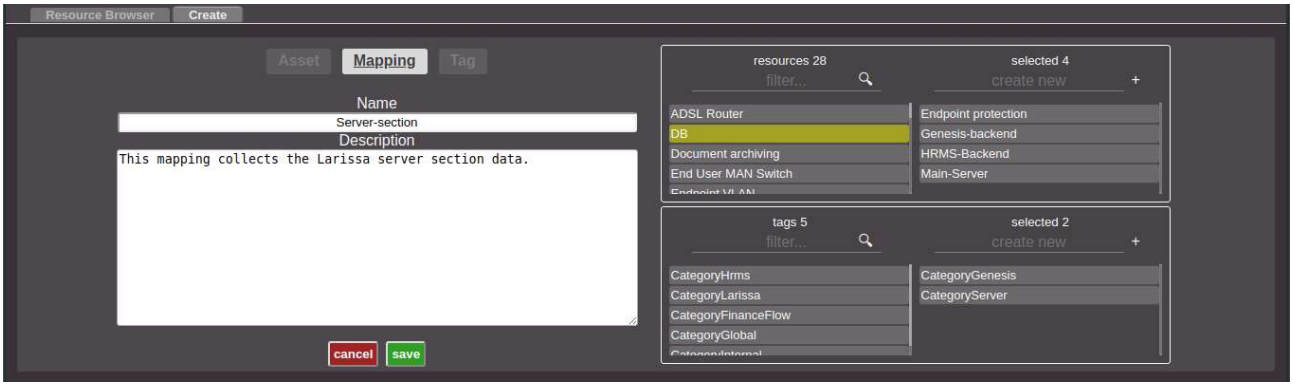


Figure 3: Creating mappings in Dependency Mapper

2.4.2 How assets are created

After the mapping has been created and the assets are created by their names, the asset information still needs to be stored in to the system. Clicking the save button of a mapping form updates the view to a detail view of the newly created mapping as shown in figure 4.

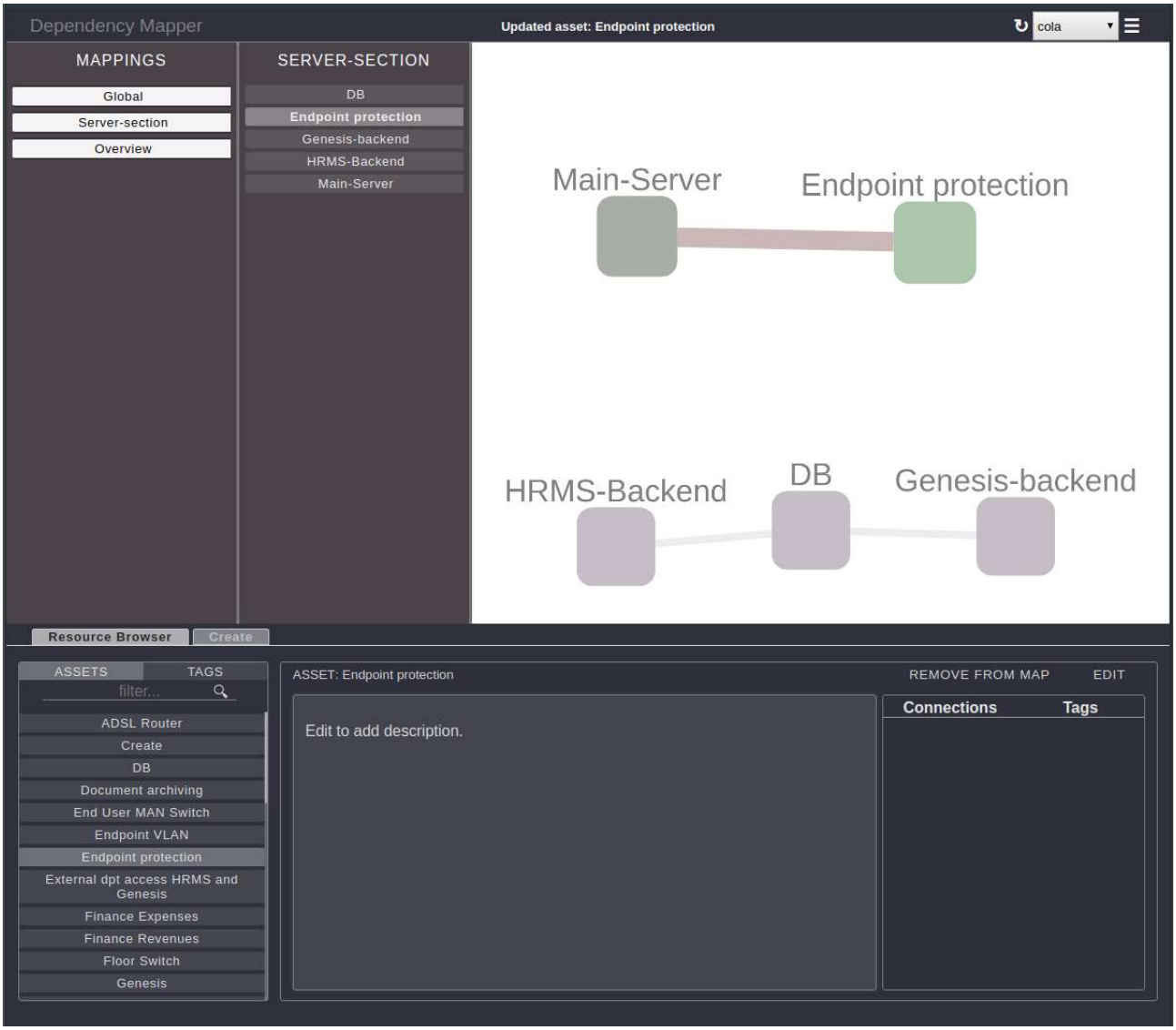


Figure 4: View after map creation

As it can be seen from the Figure 4, "Endpoint protection" is lacking the connections presented in the Table 2 and the description. At this stage after the asset is selected by clicking the item, it can be edited by opening the form from the upper right corner of the detail view.

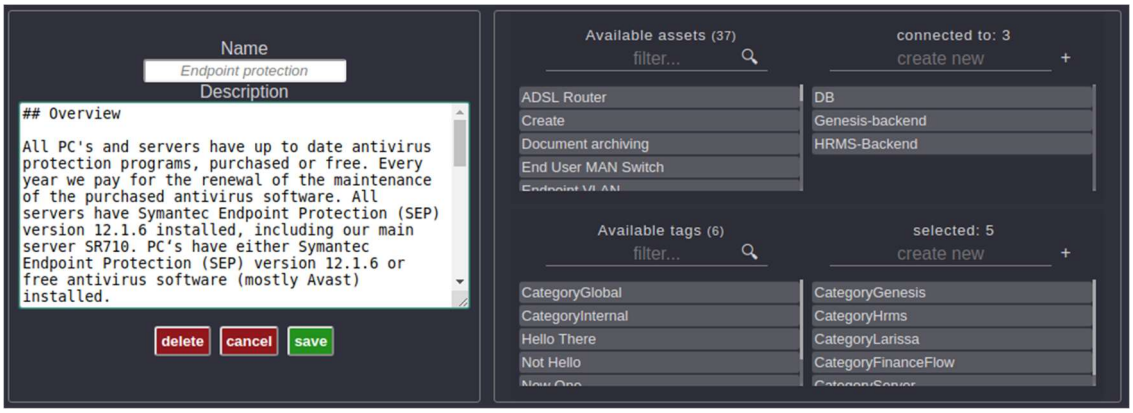


Figure 5: Asset editing

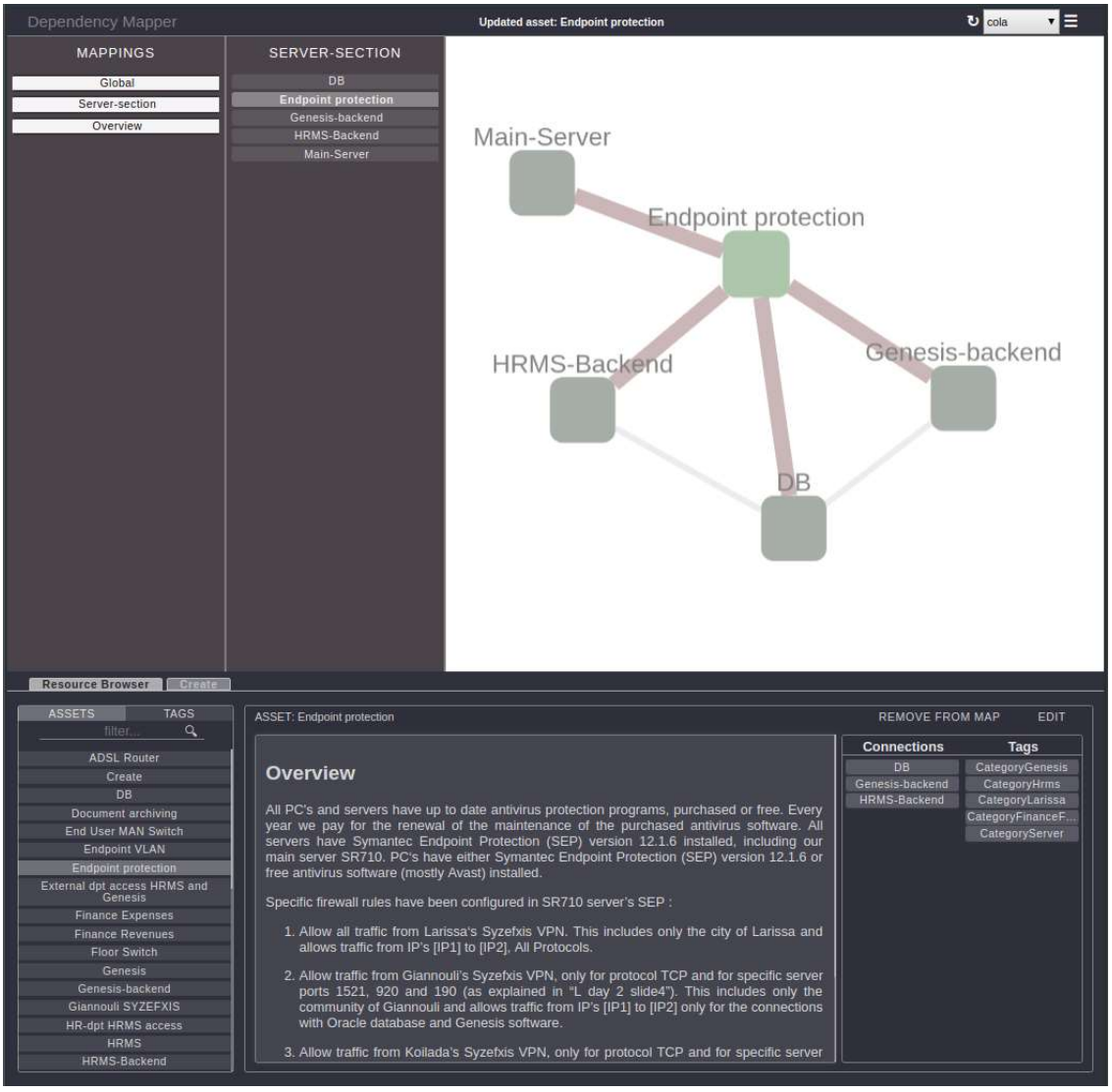


Figure 6: Updated view

Now it can be seen that the graph drawn from the "Server-section" mapping reflects the changes done in the "Endpoint protection" and that the detail view shows the information. If the needed assets are not yet in the system this view can be used to add the required assets. The details of these can be filled in later. The resulting view after clicking the Save button is shown in Figure 6.

3 Relevance to GDPR

This software merely uses data gained during the dependency analysis in WP2. Any GDPR related issues are addressed in there.

4 Further work

We are working in with our colleges at the University of the Vienna to improve the features of the Dependency Mapper. Specifically, they suggested the following additional features:

1. Node shape can be selected per each node individually.
2. Node color can be selected per each node individually.
3. Ability to group nodes visually
4. Edge labelling
5. Improving the JSON export

Figure 7 presents a design of implementation for the feature requests.

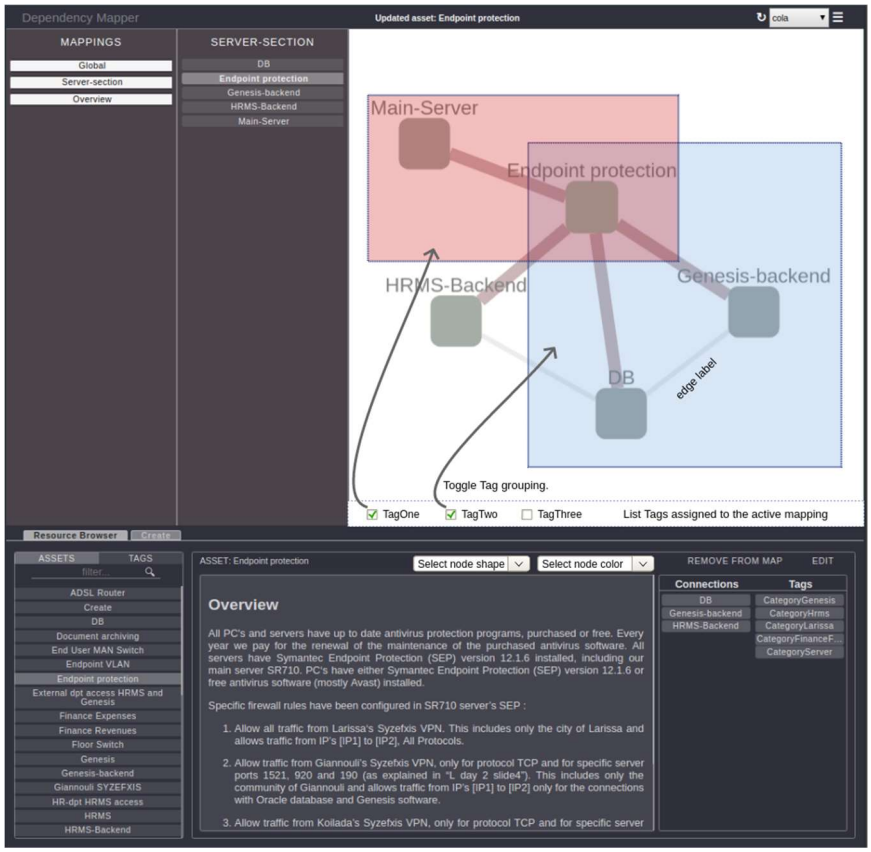


Figure 7: Design for requested features

Shape and color selection for the nodes are located in the detail view, where each individual assets color and shape can be selected. Visual grouping would utilize existing tagging system by enabling group visualization toggling based on the tags assigned to the active mapping. We will develop these additional features in Q1 and Q2 of 2019.

References and related Documents

R1 Pekka Pietikainen, Kati Karjalainen, Juha Röning, Juhani Eronen: ” Socio-technical Security Assessment of a VoIP System” in Fourth International Conference on Emerging Security Information, Systems and Technologies, Venice, 2010.

D2.4 CS-AWARE Framework

D7.3 CS-AWARE deliverable D7.3 Requirement Nr. 5 – Data Handling.

D7.4 CS-AWARE deliverable D7.4 Requirement Nr. 8 – Secondary use of personal data.

[1] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Available online: http://data.europa.eu/eli/reg/2016/679/oj .