MSCA-RISE - Marie Skłodowska-Curie Research and Innovation Staff Exchange (RISE)

This project has received funding from the European Union's H2020-MSCA-RISE-2018 programme under grant agreement No 823997.

Security ECONomics service platform for smart security investments and cyber insurance pricing in the beyonD 2020 netwOrking era

SECONDO

WP4– Cyber Security Investments and Blockchain
Deliverable D4.1 "Econometrics"

Editor(s):	CUT
Author(s):	Marco antonio Rodriguez flores (CUT), Petros Papagiannis (CUT), Nikolaos Chondrogiannis (CRO), Aristeidis Farao (UPRC), Panagiotis Bountakas (UPRC)
Dissemination Level:	**P**
Type:	**R**
Version:	**1.2**

Project Profile	
Contract Number	823997
Acronym	SECONDO
Title	Security ECONomics service platform for smart security investments and cyber insurance pricing in the beyonD 2020 netwOrking era
Start Date	Jan 1st, 2019
Duration	48 Months

Partners

	University of Piraeus Research Center	Greece
	UNIVERSITY OF SURREY	United Kingdom
	Cyprus University of Technology	Cyprus
	UBITECH LIMITED	Cyprus
	LSTech Espana	Spain
	Cromar Insurance Brokers LTD	Greece
	Fogus Innovations & Services P.C.	Greece

Document History

VERSIONS

Version	Date	Author	Remarks
0.1	11/01/2021	CUT	Table of contents
0.2	5/02/2021	CUT/UPRC	CAPEC Cyber Attacks
0.3	19/02/2021	CUT/UPRC	CIS Controls
0.4	22/02/2021	CUT/UPRC	ECM Tool
0.5	25/02/2021	CUT/UPRC	Introduction, Conclusions and Future Work
1	26/02/2021	CUT/UPRC	First draft
1.1	02/02/2021	CUT/UPRC	Internal Review
1.2	12/03/2021	CUT/UPRC	Final version

Executive Summary

A particular aim of the SECONDO project is to design, analyse and implement a Cyber Security Investment Module (CSIM). One of its most pillar components, is the Econometrics Module (ECM). In particular, it will provide estimates of all kinds of costs of potential attacks and it will take into account costs, (i.e., purchase, installation, execution, etc.), of each possible security control using a set of existing econometric models. Deliverable D4.1 “Econometrics” is dedicated to the documentation of the work carried out in the first task in WP4 “Cyber Security Investments and Blockchain”, which is detailed below:

- **Task 4.1**: Cyber security cost-benefit analysis (CUT, M14-M26). This task will establish the methodology that interprets the output of a risk assessment as a proper input to cost estimation of a cyber insurance contract. The calculation of the correct premium requires a quantification formula that is bound to a methodology. This task will provide a description of such a methodology. It will finally deliver the Econometrics Module (ECM) that provides estimates of all kinds of costs of potential attacks as well as costs, (e.g., purchase, installation, execution), of each possible security control, (i.e., Technical, organisational, procedural, etc.), using a set of existing econometric models.

Table of Contents

Table of Figures

Table of Tables

Table of Equations

1 Introduction

This section provides a brief overview of one of the most important objective and task of the SECONDO project. It further provides a description on the scope of the task carried out in this deliverable together with a summary of the future tasks and objectives to meet the proposed outcome of the project.

1.1 Role of the Deliverable

The aim of this deliverable is to provide a brief overview of the methodology and the different technologies used to develop and deliver the ECM of the SECONDO platform. In short, it provides information related to Task 4.1 that deals with the Cyber security cost-benefit analysis that establishes the methodology that interprets the output of a risk assessment as a proper input to cost estimation of a cyber insurance contract. In order to calculate the correct premium, we first need to define a methodology to quantify the costs of potential attacks as well as the costs of different security controls to mitigate them.

1.2 Relationship to other Deliverables

- **D3.1: Pricing Methods and Risk Modelling** – The Econometrics Module utilizes as its main input the output of the RAOHM including assets, number of employees and their corresponding risk.

- **D5.3: Decision Support for Cyber Insurance** – The Cyber Insurance Coverage and Premiums Module (CICPM) that will provide insurance exposure assessment and will estimate insurance coverage and premiums based on the insurance policy of the underlying insurer, will have as main input the output of the ECM.

- **D3.2: Big Data Collection and Processing –** The Big Data Collection and Processing Module (BDCPM) that will acquire risk related data either from internal organizational sources, e.g., network infrastructure, Security Information and Event Management (SIEM), log files, users’ interaction, etc., or external sources, e.g., social media and other internet-based sources, including Darknet, using specialized crawlers, will be as an extra source input of the ECM.

- **D4.2: Continuous Risk Monitoring and Blockchain** - The Continuous Risk Monitoring Module (CRMM) will ensure that changes on the ontological level, e.g., new threats or updates on digital assets and risk priorities are propagated to the CSIM, and will assess on a continuous basis the risk levels, including the performance of the implemented cyber security controls, allowing for the adaptation of cyber insurance contracts to changing organizational environment and the evolving cyber threat landscape. Its result will be another additional input source for the ECM.

1.3 Structure of the document

In the following sections we provide more details related to the methodology that ECM follows and implementation details for each component parts of the ECM Module. To be more specific, in Section 2 we provide details related to the Common Attack Pattern Enumeration and Classification (CAPEC) catalog of common cyber-attack patterns and how ECM utilizes them. In Section 3, we discuss the

Center of Internet Security (CIS) Controls -- a recommended set of actions for cyber defense that provide specific and actionable ways to stop today's most pervasive and dangerous attacks -- and how ECM module utilizes them to estimate cyber-attacks mitigation costs. Next, Section 4 provided details of the ECM module implementation. Finally, Section 5 concludes the deliverable.

2 CAPEC Cyber Attacks

The Common Attack Pattern Enumeration and Classification (CAPEC) [1] is a community-developed list of common attack patterns along with a comprehensive schema and classification taxonomy. As for attack patterns, they are descriptions of common methods for exploiting software systems, which is becoming increasingly common in today’s world of information security as malicious individuals and their associated actions are constantly seeking to exploit vulnerabilities in software development.

2.1 CAPEC brief description

The *Attack Patterns* provided by CAPEC are basically a detailed description of the most common attributes and approaches employed by adversaries to exploit known weaknesses in cyber-enabled capabilities. Attack patterns define the challenges that an adversary may face and how they go about solving it. They derive from the concept of design patterns applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples.

Each attack pattern captures knowledge about how specific parts of an attack are designed and executed and gives guidance on ways to mitigate the attack's effectiveness. The Attack patterns can help those developing applications or administrating cyber-enabled capabilities to better understand the specific elements of an attack and how to stop them from succeeding.

CAPEC was established by the U.S. Department of Homeland Security [2] as part of the Software Assurance (SwA) strategic initiative of the Office of Cybersecurity and Communications (CS&C) [3]. Initially released in 2007, the CAPEC List continues to evolve with public participation and contributions to form a standard mechanism for identifying, collecting, refining, and sharing attack patterns among the cybersecurity community.

Below we provide a short list of well-known attack patterns included in the CAPEC catalog:

- HTTP Response splitting (CAPEC-34)
- Session Fixation (CAPEC-61)
- Cross Site Request Forgery (CAPEC-62)
- Injection (CAPEC-66)
- Cross-Site Scripting (CAPEC-63)
- Buffer Overflow (CAPEC-100)
- Clickjacking (CAPEC-103)
- Relative Path Traversal (CAPEC-139)
- XML Attribute Blowup (CAPEC-229)

2.2 ECM CAPEC Utilization

The CAPEC catalog is available in different data formats including ZIP, XML, XSD and HTML. For the

SECONDO purposes and our use case for this deliverable we selected to work with the XML format since is the most comprehensive format that can be easily transform into different type of data structures across different programming languages that ECM module is utilizing. Towards that end, we first develop a python parser (CAPEC_XML_Parcer.py) that transforms the different XML files into a list of JSON object that can be easily processed within the ECM core code base. The python parser is able to pre-process the following CAPEC catalogs [4]:

- Mechanisms of Attack (1000.xml)
- Domains of Attack (3000.xml)
- Mobile Device Patterns (553.xml)
- Comprehensive CAPEC Dictionary (2000.xml)
- Meta Abstractions (282.xml)
- Standard Abstractions (283.xml)
- Detailed Abstractions (284.xml)
- Deprecated Entries (483.xml)

Later, in Section 4 we will provide more details on how the ECM module utilizes the CAPEC catalog to estimate the costs of different cyber-attacks.

3 CIS Controls

The Center for Internet Security (CIS) Top 20 Critical Security Controls [5] (previously known as the SANS [6] Top 20 Critical Security Controls), is a prioritized set of best practices created to stop the most pervasive and dangerous threats of today. The controls were originally developed after the US government experienced a major data loss in 2008. In the following sub-section, we provide more details related to the CIS Controls structure, the different tiers and maturity levels of each CIS Control.

3.1 CIS Controls details

In the current version of the 20 CIS Controls, three tiers have been established (Implementation Group 1 to 3) with tier 1 controls covering the basics and tier 3 organizations expected to implement all controls. In more details the 20 CIS Controls are broken down into the following:

1. **Basic** (6 Controls): These include the most cost-effective actions and focus on inventory across the network. The network core is a top priority, and individual devices and workstations are necessary to account for. That includes the *must do* for every organization that cover about 85% of risks. The following controls are included:
   1.1. **CIS Control 1 - Inventory and Control of Hardware Assets:** A comprehensive view of the devices on your network is the first step in reducing your organization’s attack surface. Use both active and passive asset discovery solutions on an ongoing basis to monitor your inventory and make sure all hardware is accounted for.
   1.2. **CIS Control 2 - Inventory and Control of Software Assets**: Another one of the top controls also deals with asset discovery, making network inventorying the single most critical step you can take to harden your system. After all, you cannot keep track of assets that you do not know you have on your network.
   1.3. **CIS Control 3 - Continuous Vulnerability Management**: Scanning your network for

vulnerabilities at regular intervals will reveal security risks before they result in an actual compromise of your data. It is important to run automated and authenticated scans of your entire environment.

1.4. **CIS Control 4 - Controlled Use of Administrative Privileges**: Administrative credentials are a prime target for cybercriminals. Luckily, there are several steps you can take to safeguard them, such as keeping a detailed inventory of admin accounts and changing default passwords.

1.5. **CIS Control 5 - Secure Configurations for Hardware and Software on Mobile Devices, Laptops, Workstations, and Servers**: Leverage file integrity monitoring (FIM) to keep track of configuration files, master images, and more. This control speaks to the need for automating of configuration monitoring systems so that departures from known baselines trigger security alerts.

1.6. **CIS Control 6 - Maintenance, Monitoring, and Analysis of Audit Logs**: System logs provide an accurate account of all activity on your network. This means that in the event of a cybersecurity incident, proper log management practices will give you all the data you need about the who, what, where, when, and how of the event in question.

2. **Foundational** (10 Controls): These are spread across several special organizational operations, and sometimes require more time, effort, and cost to implement properly. Expertise is often required to determine what are the right items for your organization to focus on. All in all, foundational provides additional protection for more sophisticated attacks, roughly another 10% of risk. The following controls are included:

2.1. **CIS Control 7 - Email and Web Browser Protections**: There are more security threats in email and web browsers than phishing alone. Even a single pixel in an email image can give cybercriminals the information they need to carry out an attack.

2.2. **CIS Control 8 - Malware Defenses**: Make sure your antivirus tools integrate well with the rest of your security toolchain. Implementing this control completely also means keeping accurate logs of command-line audits and DNS queries.

2.3. **CIS Control 9 - Limitation and Control of Network Ports, Protocols, and Services**: Control 9 implementation will help you reduce your attack surface by way of tactics like automated port scanning and application firewalls.

2.4. **CIS Control 10 - Data Recovery Capabilities**: Are you performing regular, automated backups? Ensuring proper data recovery capabilities will protect you from threats like ransomware.

2.5. **CIS Control 11 - Secure Configuration for Network Devices, such as Firewalls, Routers, and Switches**: Network devices can be secured using multi-factor authentication and encryption—just two of the many steps covered in control 11 benchmarks.

2.6. **CIS Control 12 - Boundary Defense**: This control deals with the way you control communications across your network boundaries. Implementing it requires using network-based IDS sensors and intrusion prevention systems.

2.7. **CIS Control 13 - Data Protection**: Control 13, despite its simple name, is one of the more complex and difficult to put into practice thanks to ongoing processes like inventorying sensitive information.

2.8. **CIS Control 14 - Controlled Access Based on the Need to Know**: By encrypting information in transit and disabling communication between workstations, you can start to limit potential security incidents that can occur when data privileges are overly lax.

2.9. **CIS Control 15 - Wireless Access Control**: The first step in implementing this control is inventorying your network’s wireless access points. From there, the control takes a deep dive into mitigating all types of wireless access risks.

2.10. **CIS Control 16 - Account Monitoring and Control**: To keep valid credentials out of hackers’ hands, you must have a system in place to control authentication mechanisms.

3. **Organizational** (4 Controls): As with any organization-wide roll outs, these are more expensive, ongoing, policy-related and testing items that ensure long-run, airtight effectiveness. Organizational goes beyond tools and tactics to make sure that all users are trained, all CIS controls are tested, and incident response is in place to address remaining risk and the fact that one can eliminate 100% of their risk. The following controls are included:

3.1. **CIS Control 17 - Implement a Security Awareness and Training Program**: Security training should be a bigger priority at most organizations, due in part to the widening cybersecurity skills gap. This control also emphasizes the need for ongoing security training rather than one-time engagements.

3.2. **CIS Control 18 - Application Software Security**: Code developed in-house needs security assessments through processes like static and dynamic security analysis to uncover hidden vulnerabilities.

3.3. **CIS Control 19 - Incident Response and Management**: This control helps you put strategies in place to plan and test for cybersecurity incidents, so you’re not left scrambling when they occur.

3.4. **CIS Control 20 - Penetration Tests and Red Team Exercises**: Regular penetration testing helps you identify vulnerabilities and attack vectors that would otherwise go unknown until discovered by malicious actors.

Apart from the three tiers described above each individual CIS Control is rated with a maturity level from 1 to 5 as follow:

Maturity Level 1 (Initial): Processes are usually ad hoc and chaotic. The organization usually does not provide a stable environment. Success in these organizations depend on the competence and heroics of the people in the organization and not on the use of proven processes.

Maturity Level 2 (Repeatable): At maturity level 2, an organization has achieved all the specific and generic goals of the maturity level 2 process areas. In other words, the projects of the organization have ensured that requirements are managed and that processes are planned, performed, measured, and controlled.

Maturity Level 3 (Defined): At maturity level 3, an organization has achieved all the specific and generic goals of the process areas assigned to maturity levels 2 and 3. At maturity level 3, processes are well characterized and understood, and are described in standards, procedures, tools, and methods.

Maturity Level 4 (Quantitatively Managed): At maturity level 4, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, and 4 and the generic goals assigned to maturity levels 2 and 3.

Maturity Level 5 (Optimizing): At maturity level 5, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, 4, and 5 and the generic goals assigned to maturity levels 2 and 3.

3.2 ECM CIS Controls Utilization

The ECM Module estimates the CIS Controls cost by utilizing additional information related to different tools that can be used to support the implementation of the different CIS Controls. To speed up the process we build on top of an existing list of tools provided by the Cyber Incident Response Team (CIRT), a Division of Homeland Security and Emergency Services (DHSES) available at the following website [7] – (CIS Controls Tool Mapping v1.1.1.xlsx).

The above file (CIS Controls Tool Mapping v1.1.1.xlsx) provides a resource which maps CIS controls to free and commercially available tools that can be leveraged during assessments and security reviews. The list includes 80 tools related to the CIS Controls that we manually annotate with additional information to be able to estimate the implementation cost of the different CIS Controls that they correspond to. Later in Section 4 we provide more details on how we estimate the different costs associated to different CIS controls.

4 ECM Tool

The Econometrics Module (ECM) is responsible to provide estimations of all kinds of costs of potential Cyber Attacks as well as costs, (e.g., purchase, implementation, maintenance), of each possible security controls, such as, technical, organizational, procedural, etc., using a set of existing econometric models. In this section we will provide more details related to the internals of the ECM Module and its implementation.

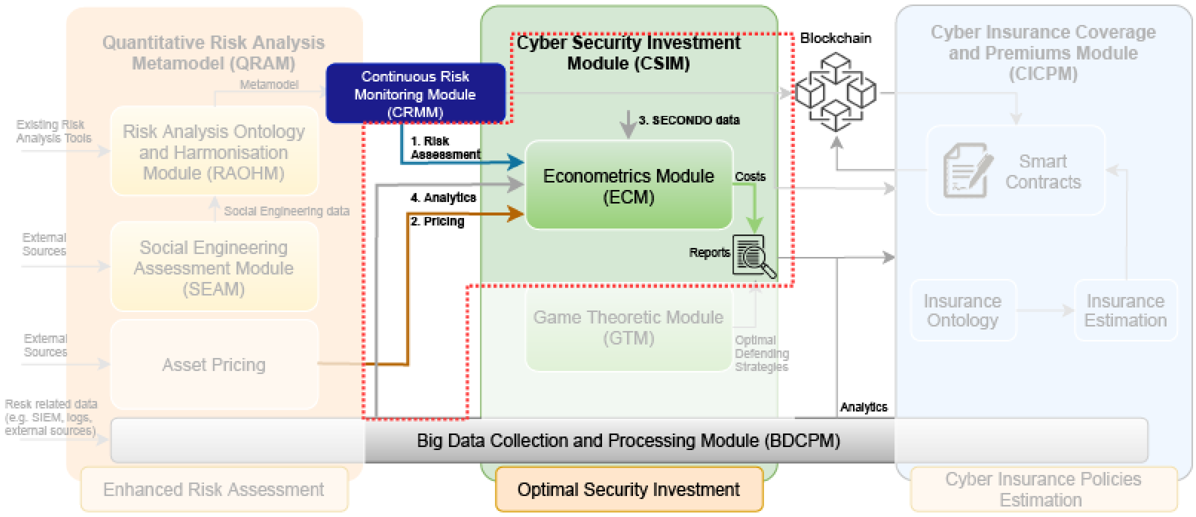

Quantitative Risk Analysis Metamodel (QRAM)
Metamodel
Existing Risk Analysis Tools
Risk Analysis Ontology and Harmonisation Module (RAOHM)
Social Engineering data
External Sources
Social Engineering Assessment Module (SEAM)
External Sources
Asset Pricing
Resk related data (e.g. SIEM, logs, external sources)
Enhanced Risk Assessment
Continuous Risk Monitoring Module (CRMM)
1. Risk Assessment
4. Analytics
2. Pricing
Cyber Security Investment Module (CSIM)
3. SECONDO data
Econometrics Module (ECM)
Costs
Reports
Game Theoretic Module (GTM)
Optimal Defending Strategies
Blockchain
Analytics
Big Data Collection and Processing Module (BDCPM)
Optimal Security Investment
Cyber Insurance Coverage and Premiums Module (CICPM)
Smart Contracts
Insurance Ontology
Insurance Estimation
Cyber Insurance Policies Estimation

Figure 1 The SECONDO high level architecture overview focused on the ECM Module and the corresponding inputs -- (1) The Risk Assessment, (2) Assets Pricing, (3) SECONDO data, (4) Big Data Analytics -- and the final output that includes the CIS Controls and Cyber Attacks cost estimations.

Figure 1 depicts and emphasizes the position of the ECM module within the overall architecture of the SECONDO project. In more details, the ECM Module accepts input from four different sub-modules as depicted at the left side of the SECONDO architecture. (1) From the **Asset pricing Module** that includes information related to the organizations tangible and intangible assets; (2) From the **RAOHM** and **CRMM** modules, ECM module receives information related to the different risk costs per organization assets; (3) **Other sources of data** as part of the SECONDO data that we already partially discuss in Section 3 and Section 2 related to CIS Controls and CAPEC Cyber Attacks catalog, respectively. And finally and (4) Data Analytics from the **BDCPM** module that provide additional information related to the organization under examination gathered from different external sources such as social network and the Web.

In the following sub-section, we provide more details related to the internal structure of the ECM module, technical details of the implementation and the algorithmic methodology that it utilizes to estimate Cyber Attacks and CIS Controls costs.

4.1 ECM Module workflow Architecture

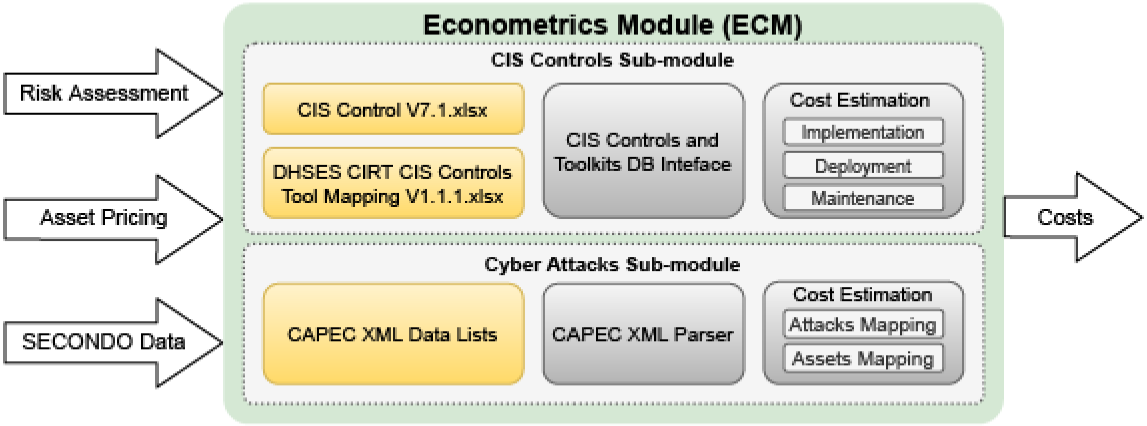


Figure 2 The architecture overview of the ECM Module.

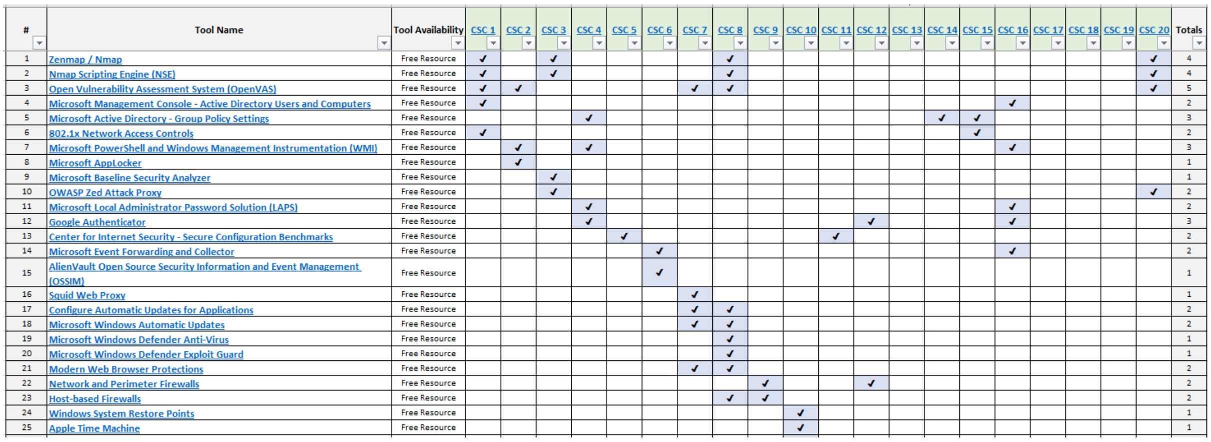
The ECM Module is sub-divided into two sub-modules. Figure 2 provides an overview of the ECM internal structure. At the top we have the component parts of the ***CIS Controls Sub-module*** that comprise the different input files (yellow colour) and the different sub-modules (grey colour). Similarly, at the bottom we have the component parts of the ***Cyber Attacks Sub-module***. The information flow is directed from left to right with all inputs from other SECONDO Modules listed at the left side of the figure and all cost estimations output of the ECM Module depicted at the right side of the figure. At the left side we have the different inputs that ECM module receives from the other component parts of the SECONDO platform: (1) **the Risk Assessment**; (2) **Assets Pricing**, (3) **SECONDO data**; (4) **Big Data Analytics** and at the right side we have the cost estimation for all the different Cyber Attacks and CIS Controls. The core implementation of the ECM Module is sub-divided into two smaller modules each undertake the responsibility for the estimation of the CIS Controls and Cyber Attacks costs, respectively. Next, we provide more details for each of those two sub-modules.

4.2 CIS Controls Sub-module

Our current implementation of the CIS Controls cost estimation sub-module is based on the CIS Controls Version V7.1 and a set of associated tools provided by the DHSES CIRT CIS Controls Tool Mapping Version V1.1.1 (see Section 3).

CIS Control cost estimation methodology: The CIS Controls cost estimation methodology is divided into three steps:

- During the first step we try to extract information related to the different costs involved to purchase, deploy, operate, and maintain each tool.
- The second step involves the pre-processing of the input data including the mapping of tools to CIS Controls as well as all inputs from the other SECONDO modules. The functionality of the first two steps is depicted in Figure 2 under the *CIS Control and Toolkit DB Interface* logic block.
- The third step involves the logic to estimate the different type of costs per CIS Control including, (1) the ***Implementation***, (2) ***Deployment*** and (3) ***Maintenance*** costs of each control. This functionality is depicted in Figure 2 under the ***Cost Estimation*** logic block.



#	Tool Name	Tool Availability	CSC 1	CSC 2	CSC 3	CSC 4	CSC 5	CSC 6	CSC 7	CSC 8	CSC 9	CSC 10	CSC 11	CSC 12	CSC 13	CSC 14	CSC 15	CSC 16	CSC 17	CSC 18	CSC 19	CSC 20	Totals
1	Zenmap / Nmap	Free Resource	✓		✓					✓												✓	4
2	Nmap Scripting Engine (NSE)	Free Resource	✓		✓					✓												✓	4
3	Open Vulnerability Assessment System (OpenVAS)	Free Resource	✓	✓					✓	✓												✓	5
4	Microsoft Management Console - Active Directory Users and Computers	Free Resource	✓															✓					2
5	Microsoft Active Directory - Group Policy Settings	Free Resource				✓										✓	✓						3
6	802.1x Network Access Controls	Free Resource	✓														✓						2
7	Microsoft PowerShell and Windows Management Instrumentation (WMI)	Free Resource		✓		✓												✓					3
8	Microsoft AppLocker	Free Resource		✓																			1
9	Microsoft Baseline Security Analyzer	Free Resource			✓																		1
10	OWASP Zed Attack Proxy	Free Resource			✓																	✓	2
11	Microsoft Local Administrator Password Solution (LAPS)	Free Resource				✓												✓					2
12	Google Authenticator	Free Resource				✓								✓				✓					3
13	Center for Internet Security - Secure Configuration Benchmarks	Free Resource					✓						✓										2
14	Microsoft Event Forwarding and Collector	Free Resource						✓										✓					2
15	AlienVault Open Source Security Information and Event Management (OSSIM)	Free Resource						✓															1
16	Squid Web Proxy	Free Resource							✓														1
17	Configure Automatic Updates for Applications	Free Resource							✓	✓													2
18	Microsoft Windows Automatic Updates	Free Resource							✓	✓													2
19	Microsoft Windows Defender Anti-Virus	Free Resource								✓													1
20	Microsoft Windows Defender Exploit Guard	Free Resource								✓													1
21	Modern Web Browser Protections	Free Resource							✓	✓													2
22	Network and Perimeter Firewalls	Free Resource									✓			✓									2
23	Host-based Firewalls	Free Resource								✓	✓												2
24	Windows System Restore Points	Free Resource										✓											1
25	Apple Time Machine	Free Resource										✓											1

Figure 3 A screenshot of the first 25 entries of the Toolkit Mapping V1.1.1.

Table 1 Toolkit directory data fields and their description.

Name	Description	URL	CIS Control(s)
The Tool name	The Tool description	The tool URL	The associated Control(s) of the tool.

Step 1 - Tools manual cost annotation: A short sample of the first 25 entries of the Toolkit Mapping V1.1.1 is provided in Figure 3 alongside the list of the top 20 CIS Controls that are mapped to. For each tool available in the Toolkit Map, we have the following information as depicted in Table 2. A short description of each field is provided below:

- **Name**: The tool name (i.e., Zenmap / Nmap).
- **Description**: A short description of the tool (i.e., Nmap Security Scanner desktop application).
- **URL**: The website URL of the tool with additional information related to the tool including purchase cost, licensing information, etc. (i.e., https://nmap.org/download.html).
- **CIS Controls**: The number(s) of the different CIS Controls that the tool can be utilized (i.e., 1, 3, 8, 20).

We use the above information to manually visit the website for each of the tools and identify the different costs related to them. In the case of zero purchase cost -- i.e., an open-source tool or freeware, not associated with any fees -- we still need to extract information related to the time required to deploy and maintain the tools and if the tool needs to be deployed per tangible or intangible asset.

Step 2 - Data pre-processing (Database Interface): During this phase, all raw data are pre-processed and normalized into a relational database schema based on Structure Query Language (SQL) standards utilizing the SQLite implementation of relational databases. We opt in utilizing the SQLite implementation for the two reasons. (1) First, SQLite is a portable implementation of relational databases allowing easy manipulation of the CIS Controls database within the overall SECONDO architecture. Second, (2) it does not pose any additional overhead to the overall SECONDO architecture to create, update and maintain the database.

Step 3 - Cost estimation logic: To estimate the overall cost of a single CIS Control *i*, the *Cost Estimation* logic block (see Figure 2 - Top right block) utilizes three equations as follow:

$$Implementatio\ Cost_i = Implementation\ Hours_i \times Employee\ Cost$$

Equation 1 Estimation of the implementation cost

$$Maintenance\ Cost_i = Maintenance\ Hours_i \times Employee\ Cost$$

Equation 2 Estimation of the maintenance cost

$$CIS\ Constrol\ Cost_i = (Purchase\ Cost_i + Implementatio\ Cost_i + Maintenance\ Cost_i)\ \times T$$

$$where\ T = \begin{cases} No.of\ items, & for\ cost\ per\ item\ (i.e.,Server, PC, Users, etc.) \\ 1, & for\ fix\ costs\ without\ individual\ license\ per\ item. \end{cases}$$

Equation 3 Calculation of the final cost

Equation 1 estimates the implementation cost by multiplying the number of hours required to implement the corresponding tool by the employee cost (average hourly wage of an employee). Similarly, Equation 2 estimates the maintenance cost by multiplying the estimated maintenance hours by the average employee cost. Finally, Equation 3, calculates the final cost of a specific CIS Control *i* by summing the three different costs (Purchase, Implementation and Maintenance) multiplied by one, if the specific CIS Control is not requiring individual license / deployment to multiple assets of the organization, or by the total number *T* of all the assets that need to be install / applied.

Implementation and Maintenance hours estimation: Both values are estimated empirically based on each tool website description. Note that the estimation of these values required some expertise in order to be estimated correctly.

Multi-pricing schemes and range pricing handling: In some cases, we have to deal with tools that are utilizing complex pricing schemes such as different tiers of pricing. For example, a tool pricing can be based on the number of licenses required (i.e., 1-49, 40-99, etc.) or based on the version of the tool that can be Pro Vs. Lite version. Later we will provide more details on how we handle such cases.

4.2.1 CIS Controls core functionality

Figure 4 depicts the Entity Relation Diagram resulting from the above process. For each SQL table depicted in Figure 4 we provide a detailed description of each field alongside all possible value(s) in the form of individual tables followed by clarification notes under each table.

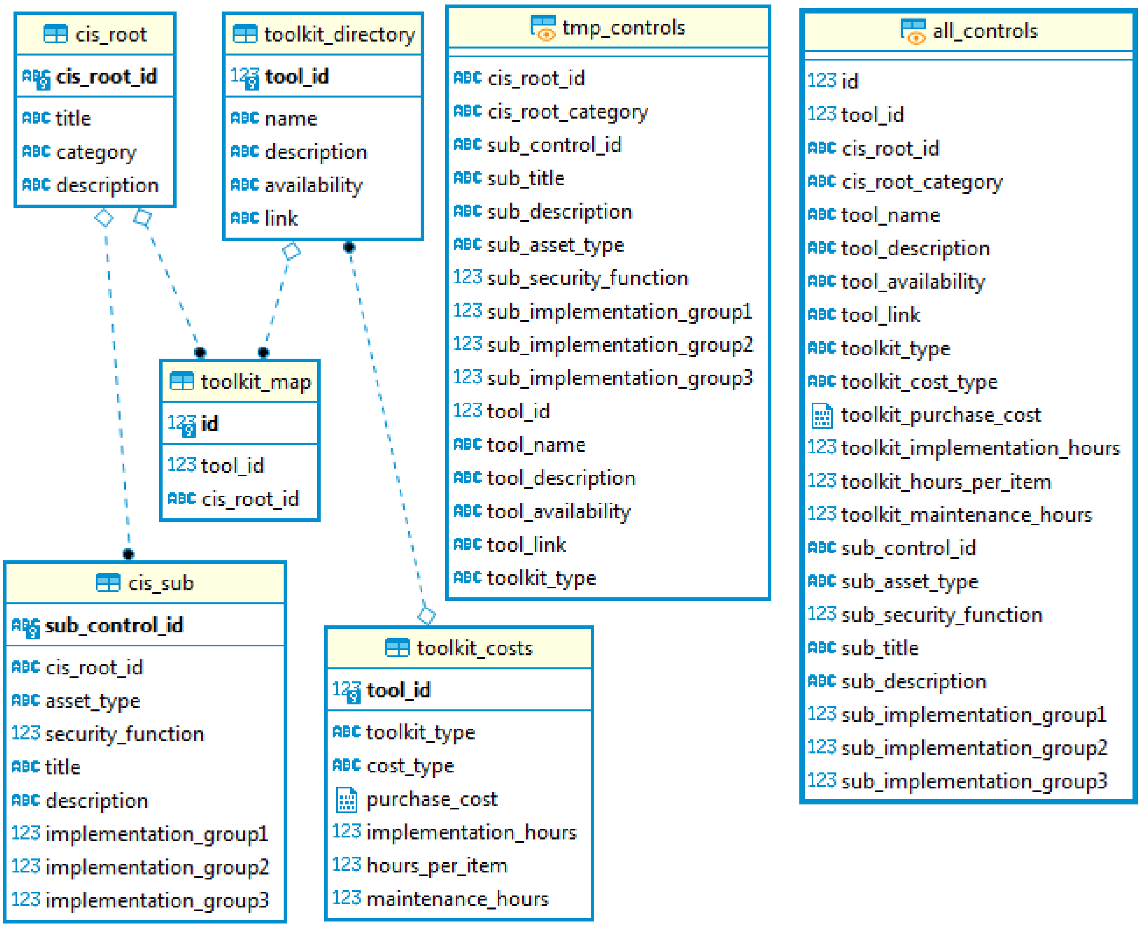


Figure 4 The CIS Controls and Tool Mapping Entity Relation Diagram of the resulting database of the *CIS Controls and Toolkit DB Interface* logic block.

Table 2 Table cis_root

Column Name	**cis_root_id**	**title**	**category**	**description**
Column Description	The Control ID [1-20]	The control title	The control category	The control description
Column Data Type	Integer	String	String	String
Column Value(s)	1 to 20	Free text	Basic | Foundation | Organization	Free text

Table 3 Table cis_sub_controls

Column Name	**sub_control_id**	**asset_type**	**security_function**	**Title**	**Description**	**Implementation group [1 to 3]**
Column description	The sub control ID [1.1-20.8]	The different asset types	The security function of the sub control	The sub control risk	The sub control description	The implementation groups(s)
Column Data type	String	String	String	String	String	String
Column Value(s)	1.1 to 20.8 depending on the CIS root control	device| application| users|	Identify| respond| protect| detect| n/a	Free text	Free text	0 or 1

		network| data| users Skill| software Dev| incidentRep| RedTest				

Columns Details for Table 3:

- **asset_type**
  - **device**: i.e., Personal Computers (PCs), Mobile Phones, Tablets, Servers, etc.
  - **application**: Different software used by the organization, i.e., Antivirus software, accounting software, etc.
  - **users**: The organization employees, i.e., the users of different PCs, mobile devices, etc.
  - **network**: CIS Control activities related to the organization network.
  - **data**: CIS Control activities related to the organization data, i.e., backup procedures, data access rules, etc.
  - **usersSkill**: CIS Control activities related to users’ skills, i.e. be able to identify fishing email, password authentication best practices etc.
  - **softwareDev**: CIS Control activities related to software development best practices.
  - **incidentRep**: CIS Control activities related to incident reporting to corresponding organizations and groups.
  - **RedTest**: CIS Control activities that required red team group activities.
- **security_function**
  - **identify**: The CIS Sub Control can identify security risks.
  - **respond**: The CIS Sub Control can respond to a security risk.
  - **protect**: The CIS Sub Control can provide protection against security risks.
  - **detect**: The CIS Sub Control can detect security risks.
  - **n/a**: The security function of the specific CIS Sub Control is not available.

Table 4 Table toolkit_cost

Column Name	**tool_id**	**toolkit_type**	**cost_type**	**purchase_cost**	**Implementation hours**	**hours_per_item**	**maintenance_hourse**
Column Description	The tool ID	The tool type	The (purchase) cost type	The purchase cost of the tool	The required hours of the tool implementation	Hours per item needed to deploy (pc, server, user, etc.)	Hours per month needed to maintain the tool.
Column Data Type	Integer	String	String	String	Float	Float	Flaot

Column Value(s)	1-80	pc| network_tool| network_pc| training_it| network_hardware| training_employees| network_task| server	float| json| json_range| json_low_high| json_per_item	Float or json object	Greater or equal to 0.0	Greater of equal to 0.0	Greater or equal to 0.0

Columns Details for Table 4:

- **toolkit_type**:
  - **pc**: The specific tool is targeted only for individual PCs.
  - **network_tool**: A networking tool that is installed once but has implementation and maintenance cost in working hours per month.
  - **network_pc**: A tool that operates on the network and needs to be configured to individual PCs. Purchase and maintenance cost can be affected by the total number of PCs in the network.
  - **training_it**: Security training activities targeting only Information Technology (IT) employees.
  - **network_hardware**: Additional network hardware (i.e., Hardware firewall, routers, etc.)
  - **training_employees**: Security training activities targeting all the employees of the organization.
  - **network_task**: A security service running on the network.
  - **server**: A software tool that targets server machines.
- **cost_type**:
  - **float**: When the tool purchase cost is a single float value.
  - **json**: When the tool purchase cost has more than two distinct values, i.e.,

```
{
  "low": 100.0,
  "medium": 200.0,
  "high": 300.0
}
```

- **json_range**: When the tool purchase cost has a cost based on a range of items. i.e.,

```
{
  "1-49": 100.0,
  "50-99": 180.0,
  "100-999": 250.0
}
```

  - **json_low_high**: When the tool purchase cost has a binary cost, i.e.,

```
{
  "low": 10.0,
  "high": 100.0
}
```

- **json_per_item**: Same as json but the purchase cost applies to all items defined by the toolkit_type column, i.e., pc, server, etc.
  - **implementation_hours**: The total hours required to deploy the toolkit. For toolkit_type *server* and *pc* we multiply by the number of assets belong to the specific category that the organization owns.
  - **hours_per_item**: Provides the hours required to deploy the toolkit on individual *pc* when the toolkit type is network_pc.
  - **maintenance_hours**: The total hours required to maintain the service per month.

Table 5 Table toolkit_directory

Column Name	**tool_id**	**name**	**description**	**availability**	**link**	**cis_map**
Column Description	The tool ID	The tool name	The tool description	The tool availability	The tool link (URL)	The tools CIS root ID mapping
Column Data Type	Integer	String	String	String	String	String
Column Value(s)	1-80	Free text	Free text	Free Resource | Commercial	URL Description i.e., Download Page: nmap.org	Comma separated integers, i.e., 1,3,8…

Columns Details for Table 6:

- **availability**:
  - **Free Resource**: The tools is free without a purchase cost (purchase_cost = 0.0).
  - **Commercial Tool**: The tool has a purchase cost.

CIS Controls cost estimation module output: The ECM - CIS cost estimation aims to module produce three different output files as follow:

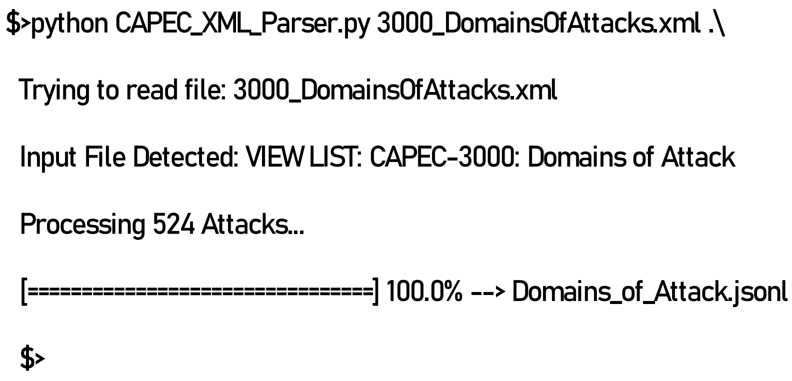
- **CIS.sqlite3**: The standalone SQLite database depicted in Figure 4.

- **CIS_costs.csv**: The CIS_costs.csv file includes a comprehensive list of all the tangible and intangible assets of the organization annotated with the joined fields of the *tmp_controls* table view and the *toolkit_costs* Table as depicted in Figure 4. The file can be used by other modules of the SECONDO platform including the Game Theory Module to estimate optimal CIS Control selection based on the estimated costs calculated by the ECM module.
- **CIS-output-raohm.csv**: The CIS-output-raohm.csv is the same as the RAOHM module output file that ECM module takes as an input, annotated with a list of all CIS Sub-Controls associated for each asset of the organization under examination. It allows the easy mapping of the organization assets between the two modules, ECM and RAOHM.

4.3 Cyber Attacks Sub-modules

As we already mention in Section 2, the ECM - Cyber Attacks costs estimation sub-module (Figure 2 - Bottom Sub-module) is based on the CAPEC Cyber Attacks catalog. The methodology to estimate cyber-attacks costs is based on a simple but effective methodology that includes a three-step process. During the first step we pre-process and normalize the different CAPEC catalog release files using the *CAPEC_XML_Parser.py* that transforms the XML file of the CAPEC attacks into a list of JSON object for easier programmatic manipulation. Next, during step 2, we utilize the normalized CAPEC attacks to manually map them to different categories of assets. Note that for each CAPEC catalog file we need to execute this step ones. Then, the resulting mapping can be utilized for multiple organizations assuming that the categorization that we utilize is generic enough to cover a wide range of assets belonging to organizations of different scales. During the step 3, we utilize the output of step 2 that includes the mapping between the different attacks and the associated asset categories that they fall in and estimate the cost of each attack based on the affected assets of the organization. Next, we provide more details related to each step mentioned above.

Step 1 - CAPEC XML Parsing: Algorithm 1 depicts a sample output of the CAPEC XML Parser processing the Domains of Attack XML file of the CAPEC catalog. The output file can then be used to manually map each of the 524 attacks into the different asset categories based on the assets input file for a specific organization. Being more specific, the input is the file *3000_DomainsOfAttacks.xml* that includes 524 unique attacks exported into a list of JSON object under the current working directory. From the aforementioned Figure 5, we can observe that the proposed tool (ECM) can effectively identify the input file and automatically generate the appropriate output filename based on the input file.



```
$>python CAPEC_XML_Parser.py 3000_DomainsOfAttacks.xml .\
 Trying to read file: 3000_DomainsOfAttacks.xml
 Input File Detected: VIEW LIST: CAPEC-3000: Domains of Attack
 Processing 524 Attacks...
 [==============================] 100.0% --> Domains_of_Attack.jsonl
 $>
```

Figure 5 A sample output of the CAPEC_XML_Parcer.py

Below we provide the details of the different fields that the parser is extracting in order to allow the categorization of a specific attack into the appropriate asset categories as we will explain during step 2. The complete manual of the CAPEC Attacks catalog schema is available at [8]. For each attack the CAPEC XML Parser extracts the following fields:

- **ID**: This field contains a unique integer identifier for the pattern. Externally, patterns will be referenced using this ID in the form *CAPEC-####* (e.g., CAPEC-12).
- **Name**: This field contains a short descriptive name for the pattern. It should be kept as short as possible but also clearly convey the nature of the attack being described.
- **Abstraction**: This field defines an appropriate abstraction level for the pattern which helps guide the appropriate information needed for its definition as well as where and how it is most usefully leveraged.
- **Description**: This field contains a detailed description of the attack including the chain of actions taken by the attacker. More comprehensive descriptions could include relevant attack trees and/or exploit graphs to elaborate this type of attack more clearly.
- **Likelihood Of Attack**: On a rough scale (Very Low, Low Medium, High, Very High), what is the overall likelihood of this type of attack typically succeeding considering the attack prerequisites, targeted weakness attack surface, skill required and resources required as well as available and likely implemented blocking solutions? The likelihood of exploit of a specific attack instance can vary greatly depending on the specific context of the target software under attack. This field is intended to capture an overall typical average value for this type of attack with the understanding that it will not be completely accurate for all attacks.
- **Typical Severity**: On a rough scale (Very Low, Low, Medium, High, Very high), what is the typical severity of impact to the targeted software if this attack occurs? The severity of a specific attack instance can vary greatly depending on the specific context of the target software under attack. This field is intended to capture an overall typical average value for this type of attack with the understanding that it will not be completely accurate for all attacks.
- **Execution Flow**: Outline of the steps involved in an attacker executing the typical flow of the attack.
- **Prerequisites**: This field describes the conditions that must exist or the functionality and characteristics that the target software must have or behavior it must exhibit for an attack of

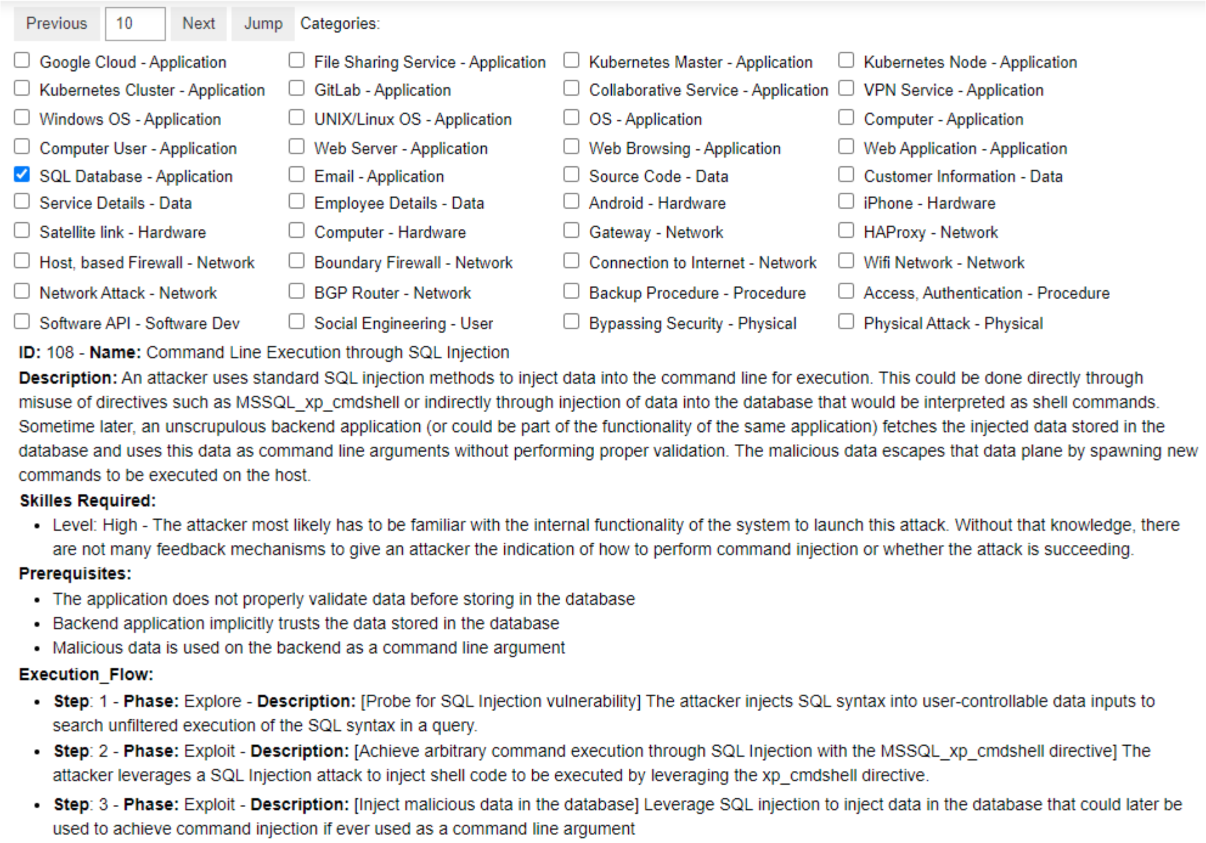
this type to succeed.

- **Skills Required**: This field describes the level of skill or specific knowledge required by an attacker to execute this type of attack. This should be communicated on a rough scale (Low, Medium, High) as well as in contextual detail. For example:
    - Low - Basic computer familiarity
    - Low - Basic SQL knowledge
    - Medium - Moderate scripting and shell experience and ability to disassemble and decompile
    - High - Expert knowledge of LINUX kernel
    - High - Detailed knowledge of target software development practices and business context (former employee)
    - Etc.
- **Resources Required**: This field describes the resources (CPU cycles, IP addresses, tools, etc.) required by an attacker to effectively execute this type of attack.
- **Indicators**: This field contains a brief description of the indicators.
- **Consequences**: What is the attacker trying to achieve by using this attack? This is not the end business/mission goal of the attack within the target context but rather the specific technical result desired that could be leveraged to achieve the end business/mission objective. In order to assist in normalization and classification, this field involves a selection from an enumerated list of defined motivations/consequences which is currently incomplete and will grow as new relevant possibilities are identified. This information is useful for aligning attack patterns to threat models and for determining which attack patterns are relevant for a given context.
- **Mitigations**: This field describes actions or approaches that can potentially prevent or mitigate the risk of this type of attack. These solutions and mitigations are targeted to improve the resistance of the target software and thereby reduce the likelihood of the attack’s success or to improve the resilience of the target software and thereby reduce the impact of the attack if it is successful.

Step 2 - Assets mapping tool: At the beginning of step 2, the cyber-attacks sub-module process the input file with the organization assets and create a list of asset categories. Next, the tool combines a predefined set of asset categories (a set of asset categories that we expect to be present in most type of organizations) with the newly create that is organization specific to include any additional categories that are not available in the predefined generic list. The resulting asset categories can then be used by the Attacks Mapping Tool to allow the manual annotation of cyber-attacks to asset categories.

In Figure 6 we present a sample screenshot of the Attacks Mapping Tool. The tool is divided into two main sections. At the top, we have the basic navigation functionality that includes three buttons, *Previous*, *Next* and *Jump* that are responsible to navigate to the previous, next and jump to a specific attack defined in the input text field located between Previous and Next buttons.

Under the navigation controls, the tool includes a list of the different asset categories. Each category is presented side by side with an appropriate checkbox to allow the user to easily select different categories that are associated with a specific cyber-attack. The generation of the asset categories is dynamic, that is, the attacks mapping tool can dynamically reconfigure the user interface to include any number of asset categories without compromising the tool functionality and visual arrangement of the asset categories. We follow the same approach in the case for the cyber-attack description fields, which are located at the bottom of the tool and we will discuss next.



□ Google Cloud - Application
□ Kubernetes Cluster - Application
□ Windows OS - Application
□ Computer User - Application
☑ SQL Database - Application
□ Service Details - Data
□ Satellite link - Hardware
□ Host, based Firewall - Network
□ Network Attack - Network
□ Software API - Software Dev
□ File Sharing Service - Application
□ GitLab - Application
□ UNIX/Linux OS - Application
□ Web Server - Application
□ Email - Application
□ Employee Details - Data
□ Computer - Hardware
□ Boundary Firewall - Network
□ BGP Router - Network
□ Social Engineering - User
□ Kubernetes Master - Application
□ Collaborative Service - Application
□ OS - Application
□ Web Browsing - Application
□ Source Code - Data
□ Android - Hardware
□ Gateway - Network
□ Connection to Internet - Network
□ Backup Procedure - Procedure
□ Bypassing Security - Physical
□ Kubernetes Node - Application
□ VPN Service - Application
□ Computer - Application
□ Web Application - Application
□ Customer Information - Data
□ iPhone - Hardware
□ HAProxy - Network
□ Wifi Network - Network
□ Access, Authentication - Procedure
□ Physical Attack - Physical

ID: 108 - **Name:** Command Line Execution through SQL Injection

Description: An attacker uses standard SQL injection methods to inject data into the command line for execution. This could be done directly through misuse of directives such as MSSQL_xp_cmdshell or indirectly through injection of data into the database that would be interpreted as shell commands. Sometime later, an unscrupulous backend application (or could be part of the functionality of the same application) fetches the injected data stored in the database and uses this data as command line arguments without performing proper validation. The malicious data escapes that data plane by spawning new commands to be executed on the host.

Skilles Required:

- Level: High - The attacker most likely has to be familiar with the internal functionality of the system to launch this attack. Without that knowledge, there are not many feedback mechanisms to give an attacker the indication of how to perform command injection or whether the attack is succeeding.

Prerequisites:

- The application does not properly validate data before storing in the database
- Backend application implicitly trusts the data stored in the database
- Malicious data is used on the backend as a command line argument

Execution_Flow:

- **Step**: 1 - **Phase:** Explore - **Description:** [Probe for SQL Injection vulnerability] The attacker injects SQL syntax into user-controllable data inputs to search unfiltered execution of the SQL syntax in a query.
- **Step**: 2 - **Phase:** Exploit - **Description:** [Achieve arbitrary command execution through SQL Injection with the MSSQL_xp_cmdshell directive] The attacker leverages a SQL Injection attack to inject shell code to be executed by leveraging the xp_cmdshell directive.
- **Step**: 3 - **Phase:** Exploit - **Description:** [Inject malicious data in the database] Leverage SQL injection to inject data in the database that could later be used to achieve command injection if ever used as a command line argument

Figure 6 Attacks Mapping tool

At the top we have a list of asset categories as defined by the list of assets of the organization under examination. At the bottom, we have the different fields that describe a specific Cyber Attack. While at the bottom of the screenshot in Figure 6 we can identify that we have the different fields that provide information for a specific cyber-attack. Due to space constrains we only depict the first few fields for a specific attack. In reality, the user can examine the attack and assign the appropriate asset categories by clicking on them and by scroll up or down can inspect all related fields in order to better understand and identify all asset categories that each cyber-attack is associated with. All fields are nicely presented to the user following appropriate indentations for list items and groups together related information to assist the user to assign asset categories as fast and accurate as possible. Note that the main goal of the tool is to minimize the time required to visually inspect all available fields related to a specific cyber-attack and identify the associated asset categories that the specific attack may affect. We repeat the above process until all attacks in the list are assign to at least one asset category.

Step 2 ends when all attacks are associated into different asset categories. The output file of this process can then be used in step 3 to estimate the costs of all cyber-attacks included in the CAPEC catalog file.

Step 3 - Attacks cost estimation logic: In step 3 we just need to estimate the total risk cost of each cyber-attack in a straight forward manner. Since from step 1 we have the list of the CAPEC cyber-attacks and from step 2 the corresponding asset categories that each attack can affect, we only need to sum up the risk cost of all affected assets of the organization under examination.

$$Attack\ Risk_i = \sum_{v=1}^{Assets} v_{risk} <=> v \in Attack_i$$

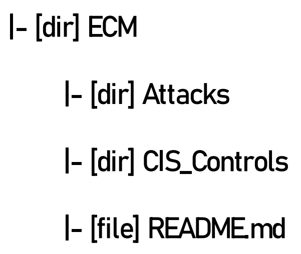
Equation 4 Summarized risk

Using Equation 4, for each Cyber Attack *i* we sum up the asset risk cost v_{risk} if and only if the asset *v* is affected by the Cyber Attack *i*. Note that a specific attack can affect more than one asset and different cyber-attacks can affect multiple assets. Since we map cyber-attacks into different asset categories, we can cover all of the above scenarios with our simple three step process described above.

Cyber Attacks cost estimation module output: The output of this module includes all the normalized JSON list files that we already pre-process with the *CAPEC_XML_Parser.py*. In addition, for each of those files, it also produces a *single .csv* file that includes all the information fields of the CAPEC Attacks catalog extracted by the *CAPEC_XML_Parser.py* annotated with the final risk cost that we calculate during step 3.

4.4 ECM Module File Structure

In this section we provide details related to the ECM module implementation file structure.



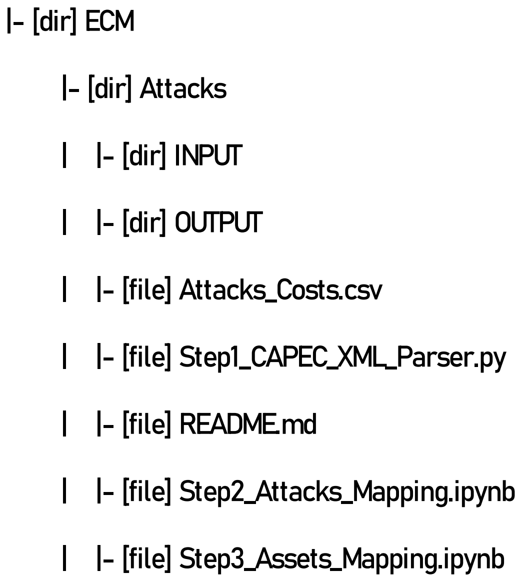
```
|- [dir] ECM
    |- [dir] Attacks
    |- [dir] CIS_Controls
    |- [file] README.md
```

Figure 7 The ECM root folder directory structure

Figure 7 depicts the root folder of the ECM module that contains two sub folders in order to estimate the CIS Controls cost and different cyber-attacks costs. The two sub-folders are named according to their functionality:

- **CIS_Controls**: Corresponds to the implementation of the CIS Controls cost estimation.

- **Attacks**: Corresponds to the implementation of cost estimation for different cyber-attacks.
- **README.md**: The **README.md** file includes a description of all the sub-folder and the functionality of the module.

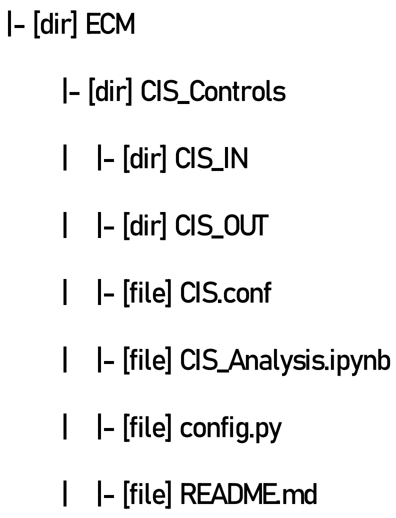


```
|- [dir] ECM
    |- [dir] Attacks
    |   |- [dir] INPUT
    |   |- [dir] OUTPUT
    |   |- [file] Attacks_Costs.csv
    |   |- [file] Step1_CAPEC_XML_Parser.py
    |   |- [file] README.md
    |   |- [file] Step2_Attacks_Mapping.ipynb
    |   |- [file] Step3_Assets_Mapping.ipynb
```

Figure 8 The ECM Attacks folder structure

Figure 8 presents the file structure of the ECM Attacks folder. Overall, the ECM Attacks folder comprise two directories and six files as follow:

- **INPUT**: The folder contains the most recent list of all CAPEC Cyber-attack catalog files (Overall 7 XML files).
- **OUTPUT**: This folder holds the output files of the manual mapping of attacks into asset categories that we manually assign during step 2.
- **Attacks_Costs.csv**: The final output file of with the attacks risk costs.
- **Step1_CAPEC_XML_Parser.py**: The CAPEC XML Parser implementation code.
- **README.md**: The *README.md* file provide additional information related to the ECM attacks implementation and how to execute the code of the sub-module.
- **Step2_Attacks_Mapping.ipynb**: The source code for the step 2 that assign asset categories to the different attacks.
- **Step3_Assets_Mapping.ipynb**: The source code of the step 3 that simply sum up the final risk costs of each attack based on the affected assets.



```
|- [dir] ECM
      |- [dir] CIS_Controls
      |   |- [dir] CIS_IN
      |   |- [dir] CIS_OUT
      |   |- [file] CIS.conf
      |   |- [file] CIS_Analysis.ipynb
      |   |- [file] config.py
      |   |- [file] README.md
```

Figure 9 The ECM CIS Controls folder directory structure

Finally, Figure 9 presents the file structure of the ECM CIS Controls folder. The folder includes two directories and four files as follow:

- **CIS_IN**: This folder groups together all the input files of the sub-module, including all the input files from other modules of the SECONDO platform and all the files related to the CIS Controls in a comma separated (csv) files.
- **CIS_OUT**: This folder holds all the output files related to the CIS Controls, such as, the final CIS Controls SQLite database file, the *CIS_Costs.csv* file and the CIS-output-raohm.csv mapping file.
- **CIS.conf**: The CIS.conf file includes all configuration variables of the sub-module as follow:
    - **DB_NAME**: (String) The filename of the final SQLite database file.
    - **UPDATE_DB**: (Boolean) We set it to True if we need to update the final SQLite database, otherwise we set it False.
    - **EMP_COST**: (Float) The average employee salary cost, required to estimate the implementation costs of the different CIS Controls.
    - **INPUT_DIR**: (String) Variable to define the directory name that holds all input files of the current module (i.e., the CIS_IN directory).
    - **OUTPUT_DIR**: (String) Variable to define the output directory of the module (i.e., the CIS_OUT directory).
    - **DEBUG**: (Boolean) We set this True if we need to print debug information to the console during the code execution otherwise, we set it False.
- **CIS_Analysis.ipynb**: The interactive implementation of the core code base of the module logic.

- **config.py**: A support python module that handles the overall module configuration file.
- **README.md**: The README.md file provide additional information related to the ECM CIS Controls implementation and how to execute the code of the sub-module.

5 Conclusions and Future Work

Deliverable D4.1 “Econometrics” presents the Econometrics Module that is a main pillar of the SECONDO platform as we have already mentioned on the previous deliverable D2.1 “Technical requirements and reference architecture” [9]. In this deliverable we presented the ECM tool and the philosophy behind the SECONDO’s approach. Moreover, we analyzed how the ECM module works internally and handles its different inputs. Moreover, we have presented a brief tutorial and its final output.

For future work, we aim to successfully develop the BDCPM module that will acquire risk related data from both internal and external organization sources. It will be an extra input source of the ECM. In addition, we will implement the Game Theoretic Module that models all possible attacking scenarios and defensive strategies, (i.e., available security controls), by employing attack graphs utilizing the results from D2.1 [9]. Finally, we will deliver the CRMM that assesses on a continuous basis the performance of the implemented risk-reducing cyber security controls allowing the adaptation of the cyber insurance contract to the changing IT environment its result is an additive input source of the ECM module.

References

[1] mitre.org, Understanding how the adversary operates is essential to effective cyber security, https://capec.mitre.org/, 2021.

[2] dhs.gov, U.S. Department of Homeland Security, https://www.dhs.gov/, 2021.

[3] cisa.gov, {Office of Cybersecurity and Communications (CS&C), https://www.cisa.gov/cybersecurity-division, 2021.

[4] CAPEC catalogs, https://capec.mitre.org/data/downloads.html.

[5] cisecurity.org, The Center for Internet Security (CIS) Controls, https://www.cisecurity.org/, 2021.

[6] sans.org, The SANS Institute, https://www.sans.org/, 2021.

[7] CIS Controls Tool Mapping, http://www.dhses.ny.gov/oct/cirt/.

[8] Complete manual of the CAPEC Attacks catalog schema, https://capec.mitre.org/documents/documentation/CAPEC_Schema_Description_v1.3.pdf.

[9] SECONDO Deliverable D2.1 "Technical Requirements, Business Cases and Reference Architecture", https://cordis.europa.eu/project/id/823997/results.