Titre: Title:	Ontologies for Insider Surveillance Indicators Rendering
Auteur: Author:	Elaheh Astanehparast
Date:	2021
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Astanehparast, E. (2021). Ontologies for Insider Surveillance Indicators Rendering [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/9901/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/9901/
Directeurs de recherche: Advisors:	Frédéric Cuppens, & Michel Gagnon
Programme: Program:	Génie informatique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ontologies for Insider Surveillance Indicators Rendering

ELAHEH ASTANEHPARAST
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Novembre 2021

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Ontologies for Insider Surveillance Indicators Rendering

présenté par **Elaheh ASTANEHPARAST**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

DEDICATION

To my companion during this journey, Pejman, for all his love and support.

ACKNOWLEDGEMENTS

The completion of this study could not have been possible without the expertise and support of the following people.

First and foremost, I would like to express my special thanks and gratitude to my supervisors Professor José M Fernandez, Professor Frédéric Cuppens and Professor Michel Gagnon for giving me the golden opportunity to be admitted in this program and supervising and leading me through this project.

In addition, I am deeply grateful to François Charest and Frédéric Michaud for their assistance at every stage of this research. Their treasured support was really influential in shaping my experiment methods and critiquing my results.

I also thank the research assistants in the lab, Jean Yves Ouattara, Marielba Urdaneta Velasquez and Militza Jean for their invaluable supervision, support and tutelage. I would like to thank my lab mate, Amine Badaoui for helping me through evaluating my project.

My appreciation also goes out to my family especially my sister, Azar, for their encouragement and support all through my studies.

RÉSUMÉ

De nos jours, les attaques croissantes de cybersécurité menacent les organisations et même les gouvernements. Au fil du temps, les chercheurs ont découvert que les menaces internes étaient plus sophistiquées que celles externes, car il y avait des privilèges plus élevés et moins d'indicateurs de compromission (IoC) à enquêter. Un autre défi important est le grand nombre d'IoC de menaces internes générés quotidiennement qui doivent être étudiés par des experts humains. Des chercheurs universitaires et des experts industriels dans le domaine ont collaboré pour proposer des solutions d'atténuation pour la prévention, la détection et l'investigation des menaces internes, même si un nombre notable de menaces internes est encore périodiquement signalé.

Parmi une variété de solutions développées pour améliorer la sécurité contre les attaquants internes, les mascarades et les acteurs non intentionnels, la modélisation ontologique peut apporter des avantages en fournissant des requêtes logiques et des relations sémantiques. De plus, les bases de données ontologiques fournissent un cadre conceptuel cohérent et une structure de données commune partagée au sein de l'organisation ou même à des niveaux supérieurs.

Cette recherche a été établie pour améliorer le processus d'enquête sur les menaces internes dans l'institution financière Desjardins dans laquelle nous visons à développer une ontologie basée sur le système IoC de menace interne défini par Desjardins qui modélise les relations sémantiques et les attaques de destruction en chaîne. Afin d'améliorer l'analyse d'IoC de menace interne en automatisant une partie du processus, nous avons visualisé les résultats des requêtes principales et fourni un tableau de bord interactif permettant à l'analyste d'interagir avec la base de données ontologique sans avoir les connaissances requises. Par conséquent, nous avons nommé notre solution "Interactive Visualization of Insider IoC Ontology" (IVIIO).

Le résultat de l'évaluation de ce projet à l'aide de trois scénarios synthétiques et de deux scénarios réalistes a démontré sa capacité à améliorer le processus d'enquête sur les menaces internes et a reçu des commentaires positifs des experts de Desjardins. Cependant, pour des résultats plus précis et fiables, des informations contextuelles et des données plus détaillées sont nécessaires. Aussi, nous pouvons demander aux analystes de travailler avec la solution lorsqu'elle sera déployée pour avoir leurs précieux retours.

ABSTRACT

Nowadays, the growing cyber-security attacks threat organizations and even governments. Through time, researchers found insider threats more sophisticated than outsider ones since there are higher privileges and less Indicator of Compromise (IoC)s to be investigated. One other significant challenge is the large number of insider IoCs generated per day that needs to be investigated by human experts. Academic researchers and industrial experts in the field have collaborated to propose mitigation solutions for insider threat prevention, detection and investigation while there is still a noticeable number of insider threats reported every so often.

Among a variety of solutions developed for improving security against insider traitors, masqueraders and unintentional perpetrators, ontological modeling can bring advantages by providing logic based queries and semantic relationships. Moreover, ontological databases provide a consistent conceptual framework and common data structure shared within the organization or even in higher levels.

This research was established to improve insider threat investigation process in Desjardins financial institution in which, we aim to develop an ontology based on the insider IoC system defined in Desjardins that models the semantic relationships and kill chain attacks. In order to enhance the insider IoC investigation by automating part of the process, we visualized the results of main queries and provided an interactive dashboard allowing the analyst to interact with ontological database without having the required knowledge. Therefore, we named our solution as Interactive Visualization of Insider IoC Ontology (IVIIO).

The result of evaluating this project using three synthetic and two observed scenarios demonstrated its ability to improve the insider threat investigation process and received positive feedback from experts in Desjardins. However, for more accurate and reliable results, contextual information and more detailed data are required. Also, we can ask the analysts to work with the solution when it is launched there to have their valuable feedback.

TABLE OF CONTENTS

DEDICATION iii

ACKNOWLEDGEMENTS iv

RÉSUMÉ v

ABSTRACT vi

TABLE OF CONTENTS vii

LIST OF TABLES x

LIST OF FIGURES xi

LIST OF SYMBOLS AND ACRONYMS xiv

CHAPTER 1 INTRODUCTION 1
1.1 Problem statement 2
1.2 Research questions 3
1.3 Scope 4
1.4 Thesis outline 4

CHAPTER 2 INSIDER THREAT 6
2.1 Basic knowledge 6
2.1.1 Insider 6
2.1.2 Insider threat 7
2.1.3 Insider threat management technologies 8
2.1.4 Indicator of compromise (IoC) 9
2.1.5 Cyber kill chain 10
2.2 Commercial solutions 11
2.3 Related works 12
2.3.1 Insider threat 12
2.3.2 Cyber-security IoC 14
2.3.3 Cyber-security kill chain 15
2.3.4 Insider threat visualization 18
2.4 Summary 19

CHAPTER 3 ONTOLOGY 21
3.1 Ontology fundamental knowledge 21
3.1.1 Formalism 21
3.1.2 Software and tools 23
3.2 Related works 23
3.2.1 Ontology design methodologies 24
3.2.2 Ontology in insider IoC 26
3.3 Justification of the methodology 28
3.4 Summary 28

CHAPTER 4 RESEARCH ENVIRONMENT AND DATA 30
4.1 Environment 30
4.1.1 Human resources overview 30
4.1.2 Insider indicator collection procedure 30
4.1.3 User interface 30
4.1.4 Insider indicator analysis procedure 31
4.2 Data 31
4.2.1 Indicators 32
4.2.2 Kill chain 33
4.2.3 Indicator event 33
4.2.4 Statistics and data analysis 34
4.3 Summary 36

CHAPTER 5 INTERACTIVE VISUALIZATION OF INSIDER INDICATOR ONTOLOGY (IVIIO) 37
5.1 System architecture 37
5.2 Ontological solution 37
5.2.1 Design 37
5.2.2 Implementation 47
5.3 Visualization 49
5.3.1 Design 49
5.3.2 Implementation 54
5.4 Summary 60

CHAPTER 6 EVALUATION 63
6.1 Assumptions 63
6.2 Scenario generation process 64

6.2.1 Phase 1: Indicator categorization 64
6.2.2 Phase 2: Indicator selection 65
6.3 Evaluation scenarios 66
6.3.1 Scenario no.1 67
6.3.2 Scenario no.2 70
6.3.3 Scenario no.3 73
6.3.4 Scenario no.4 77
6.3.5 Scenario no.5 82
6.4 Limitation 84
6.5 Summary 85

CHAPTER 7 CONCLUSION 88
7.1 Summary of the work 88
7.2 Discussion 89
7.3 Limitations 93
7.4 Future work 94

REFERENCES 96

LIST OF TABLES

Table 2.1 Top three insider threat related products based on Gartner peerInsight financial organizations ratings [1] 11
Table 4.1 Desjardins insider threat kill chain 33
Table 4.2 Top 10 users with highest risk factor 36
Table 5.1 SPARQL queries 41
Table 5.2 Subqueries for Q4 43
Table 5.3 Drivers defined for Q1 and Q2 45
Table 5.4 Drivers defined or modified for Q3 45
Table 5.5 Driver modified for Q4 45
Table 5.6 Driver modified for Q5 45
Table 5.7 Translation rule T001 46
Table 5.8 Visualization goals in the first iteration 51
Table 5.9 Visualization chart types 53
Table 5.10 Visualization goals in the last iteration 55
Table 6.1 Indicator categories defined by the scenario design team 65
Table 6.2 Description of IoCs generated by the suspicious user in scenario no.1 71
Table 6.3 Investigating the IoCs generated by the suspicious user in scenario no.2 73
Table 6.4 Description of IoCs generated by the suspicious user in scenario no.4 80
Table 6.5 Description of behavioral IoCs generated by the suspicious user in scenario no.4 81
Table 6.6 Summary of scenarios evaluation 85

LIST OF FIGURES

Figure 2.1 Pyramid of pain model (http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html) 10
Figure 2.2 Data sources and related insider detection and prevention methods [2] 13
Figure 2.3 Lockheed Martin cyber security kill chain [3] 15
Figure 2.4 Insider threat detection kill chain proposed by FBI 16
Figure 2.5 Mandiant's attack life cycle [4] 17
Figure 2.6 Kill chain proposed by Bryant *et al.* [5, 6] 18
Figure 3.1 Steps of ATOM 25
Figure 4.1 Desjardins insider IoC investigation environment 31
Figure 4.2 Distribution of indicator event sources 32
Figure 4.3 Distribution of indicators categories 34
Figure 4.4 Total number of indicator events per date 35
Figure 4.5 Histogram of number of indicator events per date 35
Figure 4.6 Total number of kill chain steps taken per date 35
Figure 5.1 IVIIO architecture 38
Figure 5.2 Description of ATOM steps 38
Figure 5.3 Intermediate query IQ1 46
Figure 5.4 Addition of translation rule T001 46
Figure 5.5 Classes and properties of the ontological model 48
Figure 5.6 Designed ontology schema 49
Figure 5.7 Security data visualization process [7] 50
Figure 5.8 Visualization chart types [7, 8] 52
Figure 5.9 Filters shared by dashboard tabs 54
Figure 5.10 A summary of the information shown in tab-1 54
Figure 5.11 Total number of behavioral and non-behavioral indicators per date by all users from 2020-11-01 to 2021-01-01 56
Figure 5.12 Top 10 most generated indicators by all users from 2020-11-01 to 2021-01-01 considering both behavioral and non-behavioral indicators 56
Figure 5.13 Top most generated steps of the kill chain by all users from 2020-11-01 to 2021-01-01 considering both behavioral and non-behavioral indicators 57

Figure 5.14 Top 10 users with highest risk factor from 2021-01-01 to 2021-03-01 considering both behavioral and non-behavioral IoCs 58
Figure 5.15 Top 10 users who have reached the highest step of the kill chain 58
Figure 5.16 Top 10 users with highest exfiltration score from 2021-01-01 to 2021-03-01 considering both behavioral and non-behavioral IoCs . . . 59
Figure 5.17 The kill chain activity of a sample user from 2021-01-01 to 2021-03-01 considering both behavioral and non-behavioral indicators . . . 59
Figure 5.18 The kill chain activity of a sample user from 2021-01-01 to 2021-03-01 versus his/her teammates 60
Figure 6.1 Top users with highest risk factor from 2020-01-01 to 2021-07-31 68
Figure 6.2 Top users reached to the highest step of the kill chain 68
Figure 6.3 Kill chain activity of "user_x" 69
Figure 6.4 Most generated IoCs and kill chain steps by "user_x" on 2021-07-15 69
Figure 6.5 Most generated IoCs and kill chain steps by "user_x" on 2021-07-15 considering only behavioral indicators 70
Figure 6.6 Top users with highest risk factor and highest step of the kill chain reached after appending information of scenario no.2 72
Figure 6.7 Kill chain activity of "user_y" 72
Figure 6.8 Top users with highest risk factor and highest step of the kill chain reached after appending information of scenario no.3 74
Figure 6.9 Kill chain activity of "user_z" 75
Figure 6.10 Total number of indicators generated by "user_z" 75
Figure 6.11 Top 10 most generated indicators generated by "user_z" . . . 76
Figure 6.12 Top 10 most generated indicators generated by "user_z" on 2021-07-09 76
Figure 6.13 User activity from 2020-01-01 to 2021-07-06 78
Figure 6.14 Most generated IoCs and kill chain steps by user from 2020-01-01 to 2021-07-06 78
Figure 6.15 Users with highest exfiltration score from 2020-05-25 to 2021-05-26 79
Figure 6.16 Most generated IoCs and kill chain steps by user from 2020-01-01 to 2021-07-06 79
Figure 6.17 Most generated IoCs and kill chain steps by user from 2020-05-25 to 2021-05-26 80
Figure 6.18 Most generated behavioral IoCs and kill chain steps by user from 2020-05-25 to 2021-05-26 81

Figure 6.19 User activity from 2020-07-15 to 2021-07-06 83
Figure 6.20 High risk users reported on 2020-12-22 83
Figure 6.21 High risk users reported on 2021-02-08 84
Figure 6.22 High risk users reported on 2021-04-11 84
Figure 7.1 :ind_1 is defined to be an instance of the class :BehavioralIndicator and is inferred to be an instance of the class :Indicator. 91
Figure 7.2 Kill chain activity is detected only when no middle step is missed. 94

LIST OF SYMBOLS AND ACRONYMS

API	Application Programming Interface
APT	Advanced Persistent Threat
ATOM	Abstractions Translation Ontology Method
BI	Business Intelligence
CAPEC	Common Attack Patterns Enumerations and Characteristics
CERT	Computer Emergency Response Team
CLM	Centralized Log Management
CPE	Common Platform Enumeration
CTI	Cyber Threat Intelligence
CVE	Common Vulnerabilities and Exposures
CWE	Common Weakness Enumeration
CybOX	Cyber Observable Expression
DL	Description Logic
DLP	Data Loss Prevention
EEP	Endpoint Protection Platform
EPP	Endpoint Protection Platform
F2T2EA	Find, Fix, Track, Target, Engage, Assess
FBI	Federal Bureau of Investigation
GCN	Graph Convolutional Network
IAM	Identity Access Management
IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
IoC	Indicator of Compromise
IVIIO	Interactive Visualization of Insider IoC Ontology
KBS	Knowledge Based Systems
MDM	Mobile Device Management
MFA	Multifactor Authentication
ML	Machine Learning
MSSP	Managed Security Services Provider
NLP	Natural Language Processing
OOPS	OntOlogy Pitfall Scanner
ORSD	Ontology Requirements Specification Document
OSI	Open Systems Interconnection

OSINT	Open Source INTelligence
OWL	Web Ontology Language
PAM	Privileged Access Management
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SAMOD	Simplified Agile Methodology for Ontology Development
SEI	Software Engineering Institute
SIEM	Security Information and Event Management
SOAR	Security Orchestration, Automation and Response
SOC	Security Operations Center
SOFIT	Sociotechnical and Organizational Factors for Insider Threat
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
SQWRL	Semantic Query-Enhanced Web Rule Language
STIX	Structured Threat Information Exchange
SWRL	Semantic Web Rule Language
SecSi	laboratoire de SÉCurité des Systems d'Information
TI	Threat Intelligence
TOVE	Toronto Virtual Enterprise
TTP	Tactics, Techniques, and Procedures
UBA	User Behavioral Analytics
UEBA	User and Entity Behavior Analytics
UEBA	User and Entity Behavior Analytics
UI	User Interface
UML	Unified Modeling Language
UP	Unified Process
UPON	Unified Process for ONtology

CHAPTER 1 INTRODUCTION

Today, computers are playing a pivotal role in human life. However, cyber-criminals are abusing this opportunity to gain access to critical and confidential data or disrupt service providers with financial, political, personal or other purposes. The Council of Economic Advisers estimated the cost of cyber-attacks on the U.S. economy in 2016, between \$57 billion to \$109 billion [9]. This example reveals the importance of empowering all businesses with effective network security solutions with the aim of detecting and preventing potential security threats to minimize losses and damages.

Among all cyber-security threats, insider attackers should be highly considered as they do not need to pass through different security defense layers to get access to the network. This condition makes insider threat detection and prevention more challenging. According to the 2018 Insider Threat Report by Fortinet [10], 68% of the organizations that participated in the survey confessed that they felt extremely to moderately vulnerable against insider threats, while only 6% announced they are well-protected. Also, they mostly believed their finance departments as their most vulnerable department, which is logically predictable. These statistics emphasize the urgent need to develop a powerful insider threat detection and prevention strategy specially in financial departments and institutions.

It goes beyond doubt that most of the work of security threat detection and prevention solutions have been done on externally-initiated attacks while insider patterns are not necessarily similar to outsiders [11]. Among all the solutions that could be used when organizations face unknown complicated insider threats, one is to have a general detection process such as a kill chain which could not be bypassed by attackers who are familiar with detection rules and predictive patterns. Kill chain steps could be defined in a general way describing every attack steps.

According to the huge number of IoCs generated by threat detection and prevention solutions, it is almost impossible for human beings to analyze the data manually. Therefore, some automated solutions are provided which lead to low detection accuracy as there are many different complicated attack patterns that could not always be translated into the machine language. To solve this problem, instead of using fully automated detection processes, data is correlated and compressed in a way that analysts can monitor and investigate suspicious activities and decide whether the IoCs are true positives or false ones. One of the most common approach that helps the analyst monitor the whole system at a glance and access the required data in real-time, is visualizing plethora of IoCs in a single and quickly digestible format [12]. Visualized trends and charts could transfer more data in less time in comparison with text logs, tables and so on. “The problem with traditional metrics is numbers and tables can be daunting and details can be missed easily. Visualizing it will enable the security team to highlight the salient points in the data” [7].

Although there are many solutions proposed to detect and prevent insider threats, varying from rule-based to machine-learning-based approaches, highly fragmented data makes sharing insider threat indicators almost impossible if a common well-defined vocabulary is not used [13]. Modeling insider IoCs using ontology could help in not only detecting insider threats but also preventing them to happen by providing sharing and analysis capabilities [13].

Insider threat identification should be considered in almost all organizations but is vital in banks and financial institutions, which are among the most potential attack targets as they have valuable sensitive data. It becomes of the utmost importance due to the fact that most of the employees in these organizations have access to confidential data [14]. In this research we are working with Desjardins group, a Canadian financial institution that has the largest presence in Quebec [15].

Desjardins insider threat investigation team uses cutting edge technology to identify potential insider threats. They also have developed their own insider threat kill chain-based on the patterns extracted by long periods of monitoring logs. Currently, DRSA operations analysts rely on a collection of more than 700 IoCs to make triage decisions and surfacing cases to prioritize for further investigation. In this research, we are going to design an ontology to model insider IoCs and use this ontology for visualization purposes in order to help analysts detect and prevent suspicious behaviors more quickly and efficiently.

1.1 Problem statement

Desjardins’s security infrastructures correlate a huge volume of logs and alerts so that the incidents could be traced and handled by human analysts efficiently. However, due to the semantic gap between different data levels, investigating the details of IoCs and incident handling, especially in a big data environment, has always been challenging [16]. Currently at Desjardins, analysts decide whether an IoC is a false positive or a true one by looking at the reports generated in a centralized IoC collector and a collection of specialised security tools. The investigation process includes querying the database about target asset sensitivity, user’s behavior and other detailed information that might help to distinguish a real threat from a normal activity which is wrongly classified as an IoC. This process is time-consuming and in some cases, complicated queries should be written to get the required information.

Also, according to the fact that the IoCs are coming from different security solutions, there are different levels of granularity.

Because of semantic modeling, ontologies are known as the best reality representation method among research community [17]. Security ontologies offer formalization of security-related concepts and their relationships. One of their objectives is to allow security professionals and processes to make faster and more accurate assessments. As Kotenko *et al.* declare, ontological approaches are becoming widely applicable to fill the semantic gaps between high-level and low-level concepts, which results in a more functional security solution [18]. Also, in terms of automating incident handling, which could effectively decrease reaction time and, consequently, data and money loss, Islam *et al.* demonstrated that ontological databases could be used to automate incident response system [19].

Having an ontological framework that summarizes and visualizes a set of indicators can improve the IoC investigation efficiency and data representability. Therefore, analysts no longer need to write and execute complicated queries as most of their required information is visualized in charts and diagrams. In order to profit from using an ontology, it should be considered that the ontological database has to be designed precisely and based on the highest prioritized tasks.

Due to the fact that there is no existing ontology applied to and evaluated in Desjardins insider threat investigation team, and based on the former discussions asserting the importance of using ontological databases in such organizations, the research team in the laboratoire de Sécurité des Systems d'Information (SecSi) at Polytechnique Montréal proposes an ontology to model insider threats in Desjardins with the aim of improving insider threat analysis time and accuracy by visualizing IoCs and their relations with users and assets in the context of Desjardin's specified insider threat kill chain.

1.2 Research questions

This research continues the work done by previous students. Sadighian, a former PhD student in SecSi lab, proposed Pasargadae comprehensive context-aware and ontology-based event correlation framework [20]. Malenfant developed an ontology design methodology for security expert systems named Abstractions Translation Ontology Method (ATOM) in his master research [21]. Finally, Ducharme presented DIOSE (*Détection d'Intrusion à l'aide d'un Système Expert basé sur l'Ontologie*[1]) [22]. We aim to develop an ontological model designed by ATOM methodology and inspired by Pasargadae and DIOSE. The main ad-

[1]Ontology-based intrusion detection using an expert system

vancement of this project is using observed data from a well-known financial institution that results in a more realistic model.

The research questions this thesis is going to answer are as follow:

1. What improvements can we make on the insider IoC investigation process using ontologies?

2. How visualizing IoCs affects the ontology-based insider threat investigation?

To answer our research questions, we used the following approach. The first phase of the work is acquiring knowledge about the previous works conducted to detect and investigate insider IoCs. For this purpose, we reviewed the works focusing on insider threat detection and visualization, cyber-security IoCs and kill chains and usage of ontological solutions in insider threat domain. We also take a look at industrial solutions provided for insider threat identification. In the second phase, we get insights on Desjardins's current processes and their special requirements in collaboration with experts and analysts in the field. Merging the results of these two phases leads us to an ontology for modeling insider IoCs in Desjardins. We are eager to add a visualization module to the solution in order to improve threat investigation process. Finally, we evaluate our approach using Desjardins's observed data and some synthetic scenarios.

1.3 Scope

Although we might have some suggestions for improving the current insider threat detection process used at Desjardins, our focus is on improving visualization to enhance investigation process as well as creating a basic ontological model to be used for future purposes as wider modeling and adding new detection and investigation solutions. Furthermore, optimizing time and space complexity of the proposed solution are out of scope of this work.

1.4 Thesis outline

In this document, a summary of the work to develop an ontology for insider IoCs as well as its deployment results in a real environment are presented. In this research we aim to design an ontology to enhance insider threat investigation process using visualization and tracking the steps of an insider threat kill chain. Therefore, the literature review is divided in six main parts: insider threat, cyber-security IoC, cyber-security kill chains, insider threat visualization, ontology design methodologies and ontology in insider IoC. The first four are discussed in Chapter 2 and the last two are reviewed in Chapter 3. Chapter 2 represents a fundamental knowledge about cyber-security and insider threats and reviews the related industrial solutions and academic works. In Chapter 3, the prerequisite knowledge of ontology as well as the current state of the art in ontology design methodology and ontology usage in insider IoC domains are presented. Chapter 4 summarizes environment specification and the general statistics of the data we had access to. The methodology details including ontology design and population process as well as visualization tasks are explained in Chapter 5. In Chapter 6, we evaluate our solution using five scenarios and provide the results. Finally, conclusion, limitations of the work and possible future improvements are discussed in Chapter 7.

CHAPTER 2 INSIDER THREAT

Since insiders could cause huge loss for organizations, many works have been done in this area from familiarizing with the context to provide detection and prevention solutions. In this chapter, we define some of the basic insider threat concepts. Afterward, the solutions available in the market as well as academic works related to insider threats, IoCs, kill chain detection and insider threat visualization on security operations are reviewed. A short summary can be found at the end of this chapter.

2.1 Basic knowledge

In this section we provide some definitions related to cyber-security and insider threat basic concepts taken from the literature.

2.1.1 Insider

Who is called an insider? The answers to this question varies among different points of view. Some emphasize on database access, some others consider having deep knowledge of network topology and so on. Reviewing the literature and combining different definitions [23–29] led us to a comprehensive definition: an insider is a person who legitimately knows about or has authorized access to organization's information system and assets physically or logically, temporarily or permanently.

Carnegie Mellon University's Computer Emergency Response Team (CERT) team defines a malicious insider as "a current or former employee, contractor, or business partner who meets the following criteria:

- has or had authorized access to an organization's network, system, or data.
- has intentionally exceeded or intentionally used that access in a manner that negatively affected the confidentiality, integrity, or availability of the organization's information or information systems" [30].

This definition only considers insiders who harm the system intentionally while there are other types of insiders who unintentionally put the organization and its resources at risk. Liu's survey categorizes insiders into three most common types: traitors, masqueraders and unintentional perpetrators [2]. Kim *et al.* define these three groups as in the following:

- “The traitor is an insider who already belongs to an organization and already has legitimate access to the organization’s resources” [31] (e.g. employees, business partners, etc.).
- “The masquerader is an insider who does not have any legal authority for the desired attack, or who has lower privileges than he wants” [31] (e.g. low-level employees, former employees, business partners, etc.).
- “The unintended insiders are who inadvertently launch attacks inside an organization due to inadvertent actions such as breaking security policy” [31] (e.g. employees, business partners, etc.).

2.1.2 Insider threat

What is called an insider threat? Based on previous researches in this area, any insider’s action that puts the system on risk could be counted as a threat. We have categorized insider threat definitions into three groups:

1. Definitions that cover both intentional and unintentional insider’s acts [24, 29, 32].
2. Definitions that refer to the insider’s act as “misuse” [33].
3. Definitions that assumes the insider’s act is intentional [13, 34].

We believe the first group demonstrates a more comprehensive schema as there have been many cases in which the insider threat is a result of an accidental mistake [35].

CERT division of Carnegie Mellon University suggested four different types of insider threats:

- “IT sabotage: an insider’s use of IT to direct specific harm at an organization or an individual.
- Theft of IP: an insider’s use of IT to steal IP from the organization. This category includes industrial espionage involving outsiders.
- Fraud: an insider’s use of IT for the unauthorized modification, addition, or deletion of an organization’s data (not programs or systems) for personal gain, or theft of information that leads to an identity crime (e.g. identity theft or credit card fraud).
- Miscellaneous: cases in which the insider’s activity was not for IP theft, fraud, or IT sabotage” [30].

Other categories (e.g. data exfiltration, data integrity or availability violation, etc.) are proposed in some resources but are not widely used [2].

2.1.3 Insider threat management technologies

Due to the high priority of managing insider threats and minimize their damage, a wide variety of technologies and procedures emerged in the market to detect and prevent insider threats. Gartner, Inc. suggests using the strategies mentioned below to actively monitor insiders [36]:

- User behavior analytics
- Physical security controls, such as facility access mechanisms
- Data Loss Prevention (DLP)
- Access privilege management
- User authorization management
- Fraud detection and escalation

Gartner Inc. also proposes a list of technologies to be used in middle size organizations to mitigate insider threats [37]:

- DLP: "A technology that performs both content inspection and contextual analysis of data flow over the network" [38].
- Endpoint Protection Platform (EPP): "A solution deployed on endpoint devices to prevent file-based malware attacks, detect malicious activity, and provide the investigation and remediation capabilities needed to respond to dynamic security incidents and alerts" [39].
- Identity Access Management (IAM): "The discipline that enables the right individuals to access the right resources at the right times for the right reasons" [40].
- Mobile Device Management (MDM): "Includes software that provides the following functions: software distribution, policy management, inventory management, security management, and service management for smartphones and media tablets" [41].

- Multifactor Authentication (MFA): "Adds a layer of protection to the sign-in process. When accessing accounts or apps, users provide additional identity verification, such as scanning a fingerprint or entering a code received by phone" [42].
- Privileged Access Management (PAM): "A solution that helps organizations restrict privileged access within an existing and isolated Active Directory environment" [43].
- User and Entity Behavior Analytics (UEBA): "Uses large datasets to model typical and atypical behaviors of humans and machines within a network" [44].

Moreover, other security solutions could also be used to monitor and detect insider threats [45]. Some of these solutions that could improve the insider threat management tasks are defined hereinafter.

- Centralized Log Management (CLM): "A comprehensive approach to network, data, and security management that uses automated tools to collect logs from across an IT infrastructure" [46].
- Security Information and Event Management (SIEM): "Technology supports threat detection, compliance and security incident management through the collection and analysis (both near real time and historical) of security events, as well as a wide variety of other event and contextual data sources" [47].
- Security Operations Center (SOC): "A centralized function within an organization employing people, processes, and technology to continuously monitor and improve an organization's security posture while preventing, detecting, analyzing, and responding to cyber-security incidents" [48].
- Security Orchestration, Automation and Response (SOAR): "SOAR refers to technologies that enable organizations to collect inputs monitored by the security operations team" [49].
- Cyber Threat Intelligence (CTI): "Evidence-based knowledge about adversaries – their motives, intents, capabilities, enabling environments and operations – focused on an event, series of events or trends, and providing a decision advantage to the defender" [50].

2.1.4 Indicator of compromise (IoC)

Haringtin defines an IoC as "a piece of information that can be used to identify a potentially compromised system" [51]. Usually, security analysts face an enormous volume of IoCs

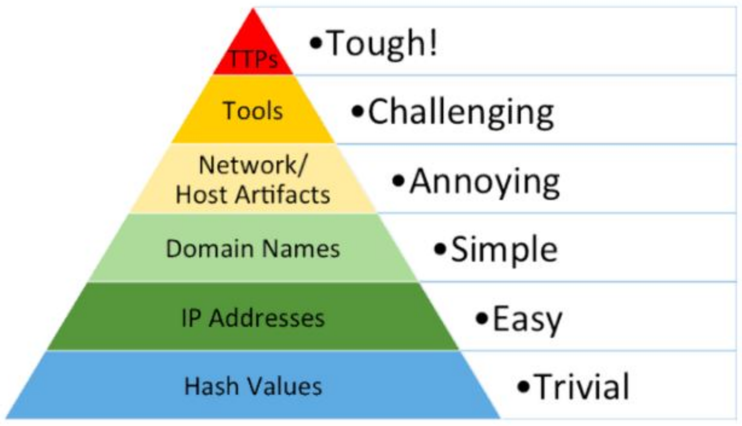


Figure 2.1 Pyramid of pain model (http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html)

including many unreliable ones. This fact makes the IoC investigation process challenging and time consuming.

Bianco developed a hierarchical schema of IoCs known as “Pyramid of Pain” [52] demonstrated in Figure 2.1 containing six levels of IoCs named: hash values, IP adresses, domain names, network/host artifacts, tools and Tactics, Techniques, and Procedures (TTP)s. As we move upward in the pyramid, the complexity of the IoCs increases and it gets harder to prevent the threats to happen. The philosophy behind the name of this model lies in the fact that the more complicated IoCs an organization is able to make use of, the more pain it will cause the adversary.

2.1.5 Cyber kill chain

U.S. military named an integrated end to end process as a chain where anyone fault will ruin the whole process [3]. The steps of this chain are known as Find, Fix, Track, Target, Engage and Assess (F2T2EA) and it works as follows: Find adversary target suitable for engagement, fix their location, track and observe, target with suitable weapon or asset to create desired effect, engage adversary and assess effects [3].

Hutchins *et al.* define an intrusion kill chain as a “systematic process to target and engage an attacker by creating desired effects” [3]. Their designed kill chain known as “Lockheed Martin kill chain” contains seven phases: reconnaissance, weaponisation, delivery, exploitation, installation, command and control (C2) and actions on objective. A traitor or an unintentional perpetrator might not need to pass all these steps as they currently have their required privileges. However, masqueraders have to take more steps to steal or somehow acquire the required privileges for attacking the system.

2.2 Commercial solutions

When it comes to commercial products, there is very restricted information about their algorithms and infrastructures. Therefore, we tried to find the cutting-edge insider threat related solutions available in the market and present them in this section. As mentioned before, there are several security tools that are suggested to be used for managing insider threats. Table 2.1 lists the top three products in DLP, CTI, Endpoint Protection Platform (EEP), SIEM, SOAR and UEBA. The ratings are based on Gartner peerInsight reviews which are provided by verified end-user professionals [1]. As the functionalities of the products in the same group are mostly the same, there was no point to comparing them.

Table 2.1 Top three insider threat related products based on Gartner peerInsight financial organizations ratings [1]

Product	First rating	Second rating	Third rating
DLP	Symantec Data Loss Prevention	McAfee DLP	Forcepoint DLP
CTI	Recorded Future Intelligence Services	PhishLabs Digital Risk Protection	IntSights External Threat Protection Suite
EEP	McAfee Endpoint Security	Symantec Endpoint Protection	Kaspersky Endpoint Security for Business
SIEM	LogRhythm	QRadar (by IBM)	Splunk Enterprise
SOAR	Siemplify	Cortex XSOAR (by Palo Alto Networks)	InsightConnect (by Rapid7)
UEBA	Varonis Data Security Platform	Advanced Threat Analytics (ATA) (by Microsoft)	Incydr (by Code42)

It also should be mentioned that since this research focuses on insider threats in financial institutions, the results shown in Table 2.1 are filtered in a way that top products selected by only financial organizations are considered and summarized.

2.3 Related works

In this section, the previous academic works focusing on insider threats, kill chains and insider threat visualization are categorized and reviewed.

2.3.1 Insider threat

Insider threats and the related mitigation actions have been studied for almost two decades, resulting in an immeasurable number of researches published. As it is unfeasible to overview all the previous works and in order to gain insight into the most of the related works possible, here we summarize the findings of the surveys in the context, each reviewing lots of previous works.

One of the first exclusive surveys of insider threat detection methods could be the one published by Salem *et al.* [53] categorizing all the previously published detection methods into three main categories: host-based user profiling, network-based sensors and integrated approaches. They derived how often each category is deployed based on the insider type. The survey reviews different machine learning and modeling solutions but only considered intentional insiders. The lack of ground truth which is introduced as the biggest challenge in the context, is believed to result in uncertain efficiency of all the proposed solutions.

Hunker *et al.* provided an overview of the insider threat definitions and mitigation techniques considering three insider threat types as: misuse of access, bypassing defenses and access control failure [34]. They divided mitigation approaches into three main categories: 1) technical, 2) socio-technical and 3) sociological, psychological and organizational approaches. To be able to tackle insider threats successfully, deploying a combination of these three categories is suggested.

A research trying to overview some specific insider threat detection methods is provided by Azaria *et al.* focusing on behavioral methods [54]. Instead of proposing a systematic literature review, the authors took a look at previous studies on psychological and social theories, anomaly detection approaches, honeypots, graph-based approaches and game theory approaches. Subsequently, they introduced their framework namely Behavioral Analysis of Insider Threat (BAIT) which is claimed to be able to analyze malicious insiders behaviors.

To the best of our knowledge, the only survey on insider threats that considers kill chain is the one prepared by Liu *et al.* [2]. They employed Lockheed Martin kill chain [3] to accommodate the steps of threat that should be taken by a masquerader. The use of kill chain is suggested to prevent the threat in its early stages. In a mapping between type of insider and kill chain steps, the only step that all insiders take is “Actions on objectives” and the rest is only mentioned to be taken by masqueraders. The authors considered three types of insiders as traitors, masqueraders and unintentional perpetrators. They categorized previous works based on the data source in three main groups: host, network and contextual data. Afterwards, they mapped each data source to the related detection method used by the researchers (Figure 2.2). For example, they found that the researches worked on Email and LDAP network logs, mostly used rule-based detection methods.

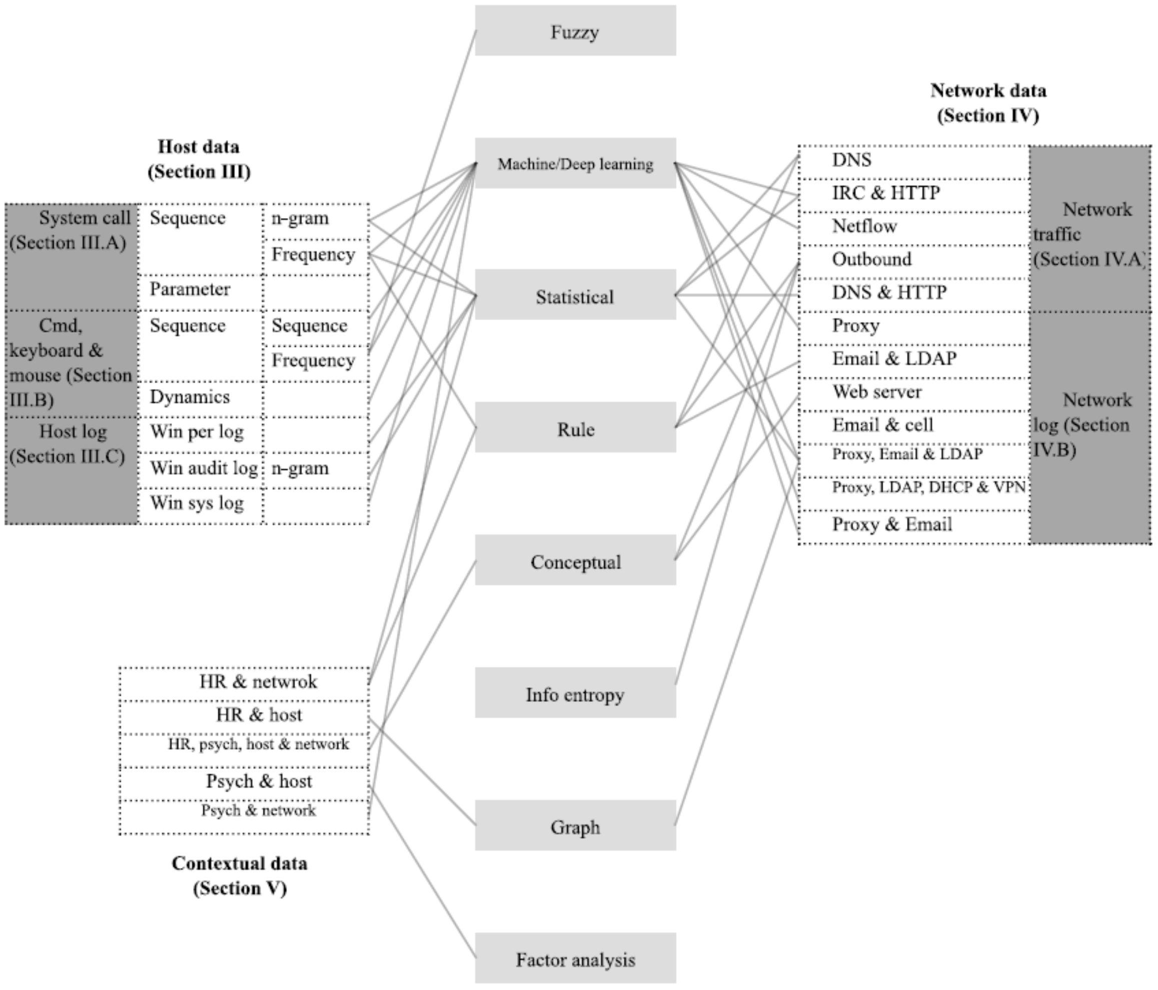


Figure 2.2 Data sources and related insider detection and prevention methods [2]

The diagram shown in Figure 2.2 is notable since it reveals the appropriate insider threat detection methods deployed based on the nature of the data source. Although there is no ontology-based method mentioned among detection approaches, we can consider graph-based category which is closer to our research domain and is linked to only two data sources: “Proxy, Email and LDAP” and “HR and Host”. Therefore, it seems graph-based methods are not widely used yet.

Homoliak *et al.* decided to systematically categorize different insider threat solutions and prepare a unified structural taxonomy for definitions [55]. Not only does this taxonomy cover insider threat concepts but also considers 5W1H[1] questions. Moreover, all the literature in the field are categorized into four main groups: 1) incidents and datasets, 2) analysis of incidents, 3)simulation and 4) defense solutions, each containing different sub-groups.

In addition to the surveys, a specific research on insider threat in financial organization is worth to be mentioned. Carnegie Mellon University's CERT has studied insider threat cases happened in banks and financial institutions to analyze their specific characteristics [56]. They ended up with six important findings:

1. Most of the incidents are not highly complicated.
2. Criminals pre-plan the attack.
3. Financial gain was the most prevalent motive.
4. No unique profile was shared between perpetrators.
5. The incidents were detected by different methods and people.
6. Threat target organizations suffered financial loss.
7. Incidents often happens in a normal place and during normal time.

They concluded the financial organizations should work on their security policies as there are a wide variety of employees threatening the system.

2.3.2 Cyber-security IoC

Lots of taxonomies related to cyber security IoCs are proposed by Mitre:

- Common Vulnerabilities and Exposures (CVE) [57] contains the identifiers for security or information vulnerabilities.
- Common Platform Enumeration (CPE) [58] contains standardized named IT products in a machine readable format.
- Common Weakness Enumeration (CWE) [59] is a complementary dictionary for CVE containing attack prerequisites, explanation and countermeasures.

[1]What, who, where, when, how and why

- Common Attack Patterns Enumerations and Characteristics (CAPEC) [60] provides a comprehensive list of patterns, descriptions and mitigation approaches related to the observed attacks exploiting vulnerabilities and weaknesses.

- Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) [61] is a knowledge base of attack tactics, techniques and tools including 14 tactics, 185 techniques, and 367 sub-techniques.

All these taxonomies facilitate the interoperability by sharing a common language but do not allow for a logic-based query and have limited search capabilities [62].

2.3.3 Cyber-security kill chain

In this part the most well-known literature focusing on insider and general cyber-security kill chains are reviewed. It is noteworthy that in some cases, the cyber-security kill chains designed for outsider attacks could be deployed for describing insiders behavior [63].

Legacy cyber kill chain (a registered trademark of Lockheed Martin) could be considered as the most prominent kill chain for cyber attacks [3]. It contains seven main steps, namely Reconnaissance, Weaponisation, Delivery, Exploitation, Installation, Command and control (C2) and Actions on objectives (Figure 2.3).

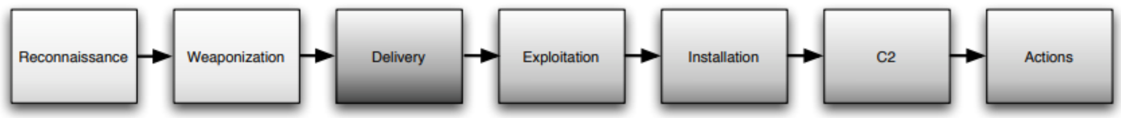


Figure 2.3 Lockheed Martin cyber security kill chain [3]

We will briefly summarize each stage in the following:

1. Reconnaissance: Research for selecting the targets using Open Source INTelligence (OSINT) or social engineering [64].

2. Weaponisation: Preparing a malicious deliverable payload named weapon (such as a file containing a remote access trojan or an exploit).

3. Delivery: Transferring the malicious weapon prepared in second step to the target selected in first step.

4. Exploitation: Exploiting the vulnerability to deliver the malicious payload to target.

5. Installation: Installing the malicious payload (such as a backdoor) to provide persistent access by attacker.

6. Command and control (C2): Giving the attacker access to the target asset using a C2 server.

7. Actions on objectives: Performing the desired malicious actions such as data access, modification, exfiltration, encryption and so on.

The Federal Bureau of Investigation (FBI) insider kill chain could be assumed to be one of the first kill chains focusing on insider threats including four steps (Figure 2.4) [65].

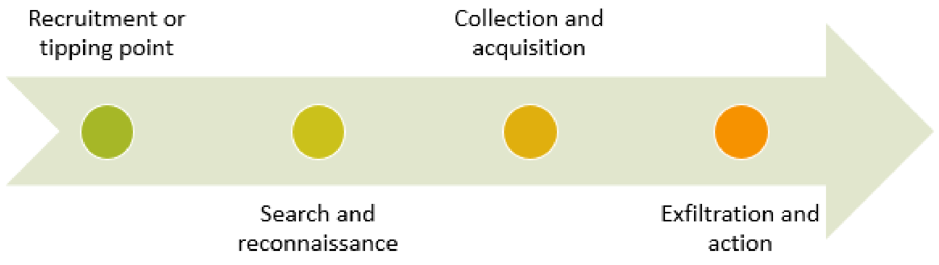

Recruitment or tipping point
Search and reconnaissance
Collection and acquisition
Exfiltration and action

Figure 2.4 Insider threat detection kill chain proposed by FBI

This kill chain was presented at BlackHat USA conference in 2013, is adopted by Tripwire [66] and revised by ZoneFox in 2015. The revised version contains an extra step named "Exploitation" placed in the middle of the kill chain [67]. The description of the steps of the revised version is provided below:

1. Recruitment or tipping point: Getting the motivation for the attack.

2. Search and reconnaissance: Seeking substantial information.

3. Exploitation: Accessing the desired asset containing the valuable data.

4. Collection and acquisition: Accessing the desired data.

5. Exfiltration and action: Exfiltrating data using different channels.

FireEye has designed a cyber kill chain named "Mandiant's Attack Lifecycle" specifically for Chinese Advanced Persistent Threat (APT) lifecycle which is reported to behave slightly different than generic attacks [4]. This kill chain includes eight steps and is iterative over four middle steps demonstrating in Figure 2.5. The descriptions of the steps are as follows.

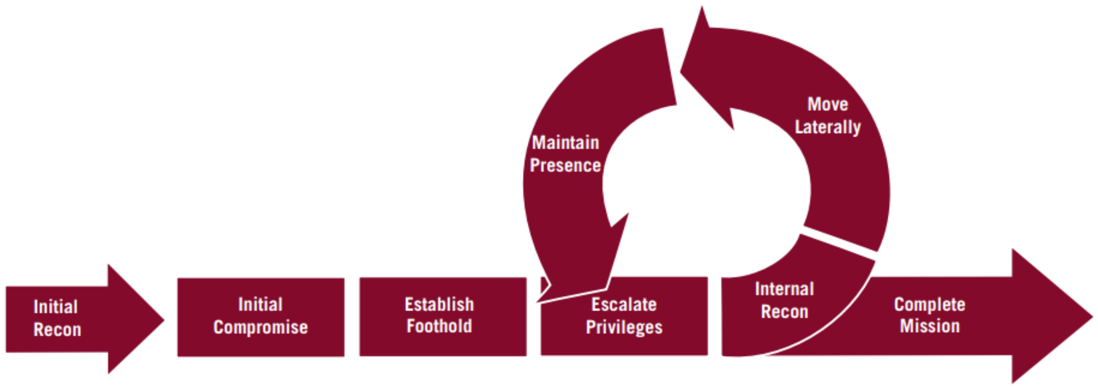


Figure 2.5 Mandiant's attack life cycle [4]

1. Initial compromise: deploying different methods (such as phishing, social engineering, etc.) to penetrate a target asset.

2. Establish foothold: Using backdoors to provide a consistent communication channel between the attacker and target assets.

3. Escalate privileges: Trying to get a higher privilege account to have access to more data and resources by cracking password hashes or using related tools.

4. Internal reconnaissance: Finding information about target environment (such as computers, users and groups in the network and so on) and clues (such as file servers, mail servers and so on) to reach data of interest.

5. Move laterally: Progressing in the target network to achieve the data of interest or compromise a system that allows to access it by gaining access to different machines in the network.

6. Maintain presence: Ensuring having persistent control over pivotal assets by installing new backdoors or different malware types that communicate with a variety of C2C servers.

7. Complete mission: Exfiltrating the files of interest in a protected way using different methods such as FTP, installed backdoors, etc.

In a two years study by Bryant *et al.* in a real Managed Security Services Provider (MSSP)'s SOC, none of the aforementioned kill chains was expressive enough to be deployed [6]. Therefore, the authors designed their own kill chain (Figure 2.6) focusing on sequencing metadata requirements so that they could be able to define strict deterministic correlation rules in LogRhythm SIEM.

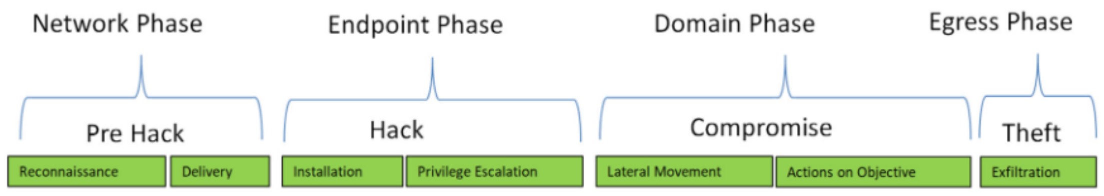


Figure 2.6 Kill chain proposed by Bryant *et al.* [5, 6]

They did not discuss what exactly happens in each phase. However, since the kill chain is inspired by Lockheed Martin and Manidant, we can substitute the descriptions provided before for each related phase. Additionally, the list of metadata they are looking for in each phase could demonstrate the description of the stage which is listed below:

1. Network phase

   Reconnaissance: Probing, enumeration

   Delivery: Host access, network delivery

2. Endpoint phase

   Installation: Host delivery, software modification

   Privilege escalation: Privilege escalation, privilege use

3. Domain phase

   Lateral movement: Internal reconnaissance, lateral movement

   Actions on objectives: Data manipulation, obfuscation

4. Egress phase

   Exfiltration: External data transfer

2.3.4 Insider threat visualization

According to the fact that most of the threat detection approaches results in a high ratio of false positive alerts, analysts are employed to investigate the threats manually. In order to improve the investigation process which takes a long time due to the massive number of alerts, some approaches can be used, among which visualization is becoming widely recognized as one of the most effective solutions in which the analysts get a big picture of what has happened without spending lots of time querying the database. Also, the interactive dashboards provide detailed explanations by just clicking or hovering on a specific data point.

SANS institute proposed five main steps for the security visualization process: 1) Visualization Goals, 2) Data Preparation phase 3) Exploration phase, 4) Visualization phase and 5) Feedback and fine-tune [7]. For each phase, a list of activities are explained in details. They also suggested which chart type is suitable to be used for the visualization based on the nature of the data.

There are plenty of researches demonstrating how visualization could improve cyber security related tasks. In this section, we will take a look at the works focused on visualizing insider threats.

Visualizing insider threats began with simple graphs encapsulating complicated work group roles and users behaviors [12]. In this approach, all user's benign activities are extracted and defined as the normal behavior of the work group role. Therefore, any anomaly is detected as a suspicious activity which may indicate an insider threat.

In another work by Legg [68], anomaly detection techniques are combined with visual analytic to monitor users activities, detect anomalies regarding the observable features of user profiles and investigate details of the activities of the suspicious user to decide whether the anomaly is related to an insider threat or not. They evaluated their approach using CMU-CERT insider threat datasets.

User Behavioral Analytics (UBA) was a tool developed by Haim *et al.* as an extension of IBM's QRadar security analytics environment [69]. This tool determines and visualizes the security risks based on users access and network usage to improve security analysis in real time. They focused on usage anomalies while the previous literature mostly worked on system and network anomalies. This tool is evaluated in financial institutions and could get positive feedback.

2.4 Summary

The first section of this chapter aims to provide relevant basic information and definitions. We defined two basic insider threat concepts that are used in this research frequently: insider and insider threat. We reviewed different definitions provided by previous works and tried to end up with the most comprehensive definition. Moreover, it has a great importance to get familiar with the technologies in this context. Finally, the IoC and cyber kill chain terms are defined and explained.

In the next section, the solutions available in the market are reviewed. Since the detailed information of commercial products are not revealed and they mostly claim to do the similar tasks, we decided to use Gartner information to find the most related products for insider threat management and provide a ranking based on verified users reviews.

Related academic works in four main categories (Insider threat, Cyber-seurity IoCs, Cyber-security kill chain and Insider threat visualization) are reviewed in section 2.3. Due to the fact that there are not much insider threat kill chains proposed by previous works, we decided to broaden the area and cover not only insider threat kill chains but also the outsider ones. Also, we mentioned the most well-known taxonomies for IoCs. Finally, we also took a look at the works focused on effectiveness of insider threat visualization on security operations. We could not find any work visualizing an ontological database with the aim of insider threat management.

According to the literature reviewed, rule based insider threat detection techniques are overly specialized which makes them not effective against unknown attacks. On the other hand, automated methods such as machine learning algorithms result in high rate of false positive events. Therefore, we decided to deploy a kill chain designed specifically for insider threats that models almost all of the insider attacks. Having such kill chain, we can detect the users who are following it and report them as suspicious users. However, most of the proposed kill chains are designed for outsider attacks. Thus, we decided to use Desjardins' specific kill chain for insider IoCs. We will monitor users' behaviors in terms of the steps of the kill chain they take. In addition, we found out that to the best of our knowledge, there is no taxonomy for insider kill chain attacks. Finally, regarding the advantages of providing visualized reports in insider threat management methods reported in literature, we deploy visualization techniques to improve the insider IoC investigation process.

CHAPTER 3 ONTOLOGY

In this chapter, the basic knowledge of ontologies required for this thesis is shortly described. The first step of developing an ontological model is to select the best design methodology. In order to do so, we take a look at some of the design methodologies proposed by researchers. Also, to get familiar what benefits ontologies can bring to insider threat domain, the related works in this context are summarized subsequently. Finally, a summary and conclusion of the findings is provided at the end of this section.

3.1 Ontology fundamental knowledge

Although the ontology concept has been used in philosophy from a long time ago, in computer science literature there is no unanimously accepted definition [70]. However, we could use the definitions like "An ontology is formal explicit specifications of shared conceptualizations" [71]. Another highly cited definition that had better be mentioned is given by Guarino [72]: "An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e. its ontological commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models" [73]. As this huge volume of explanations could not be used as a definition, researchers tried to bring a simpler definition: "ontology being equivalent to a Description Logic knowledge base" [74] in which, description logics are formal knowledge representation languages that are used to produce a structured and comprehensible knowledge representation of application domains [75].

The base of an ontology is built using three main concepts: Classes, Properties and Individuals. The conceptual entities are modeled as Classes. The relationships between Classes, between two entities or between an entity and a literal are modeled as Properties. Finally, the individuals are the real instances of the classes.

3.1.1 Formalism

In recent decades, different languages such as Resource Description Framework (RDF), Resource Description Framework Schema (RDFS), Web Ontology Language (OWL), have emerged for developing ontologies. Also, query languages like SPARQL Protocol and RDF Query Language (SPARQL) and Semantic Query-Enhanced Web Rule Language (SQWRL) are defined to query the ontological database in the same way that Structured Query Language (SQL) is used to query relational databases. Finally, there is always a need to define restrictions. Value constraints and cardinality constraints are provided as property restrictions in OWL. Semantic Web Rule Language (SWRL) is emerged to cover OWL property restriction limitations and provide stronger semantic foundation by allowing the user to define rules. RDF, SPARQL and SWRL are explained in more details in the following and will be used in the next chapters.

Resource Description Framework (RDF)

“RDF is a language for representing information about resources in the World Wide Web” [76]. One can summarize developing an ontological model into two main phases: designing the conceptual model and populating the ontology with instances.

Conceptual modeling: At first, high-level concepts and their relationships should be defined. RDF tries to describe statements using triples:

$$< object >< predicate >< subject >$$

<object> is the thing we want to state something about. <predicate> is the property or characteristic of <subject> and <object> is the value for <property> [76]. For example the statement “an indicator event is generated by a sensor” is represented by the tuple:

$$< IndicatorEvent >< isGeneratedBy >< Sensor >$$

One could assume subject and object as two nodes in a graph and predicate as the arc connecting these two nodes. All the constrains and restrictions are defined in this phase.

Population: Once the empty model is defined, instances of the classes are used to populate the ontology. Each instance can be defined by its type and its relationship with other instances and literals.

$$< sensorInstance01 >< rdf : type >< Sensor >$$

In the example above, an instance of the class Sensor is defined.

Semantic Web Rule Language (SWRL)

SWRL allows reasoning by defining rules on concepts and individuals. A SWRL rule consists of two main parts: antecedents and consequents. If conditions of antecedents are met, consequents will be applied. For instance, when there is no direct relationship between the Event class and Sensor class, a rule to find the source of an event could be written as below:

$$Event(?e) \wedge hasIndicator(?i) \wedge Indicator(?i) \wedge isGeneratedBy(?i, ?s) \implies hasSource(?e, ?s)$$

SPARQL

One other semantic web technology is SPARQL, which is used to query the ontology and can play the role similar to SQL in relational databases. SPARQL produces the desired outcome by filtering the data based on some constraints. To give an example, the SPARQL query to retrieve the top 10 users who have the highest risk factor would look like the query below:

```
PREFIX : <http://www.ontology.ca/ontology-1.0#>
select ?userName ?risk where {
  ?user a :User;
   :hasUserName ?userName;
   :hasUserRiskFactor ?risk
}
ORDER BY DESC(?risk)
limit 10
```

3.1.2 Software and tools

There are many different technologies and tools such as Protégé [77], Apache Jena [78], SWOOP [79], GraphDB [80], Virtuoso [81], Ontolingua [82], Stardog [83] to provide a user-friendly environment to design, develop, populate and query an ontological model.

3.2 Related works

Since we aim to develop an ontology for insider threats, a design methodology is needed to be followed at the first step. Also, we need to get familiar with previous works that are focused on using ontology in the domain of insider threat. This section provides the state of the art of ontology design methodology as well as ontology usage in insider threat domain.

3.2.1 Ontology design methodologies

In order to select the best methodology for designing the ontology, some of the most well-known methodologies are reviewed in the following. Acoording to the literature, there are two main roles collaborating to develop the ontology:

1. Domain Expert: The experts in the domain the ontology will be used in.
2. Ontology Engineer: The ontology experts that are responsible to model what Domain Experts illustrate.

In some cases, "Knowledge worker" is another role involved in the ontology life-cycle and will be responsible to verify the model mostly in real environment conditions. However, this could be done by domain experts as well [84].

According to the related works done from 1990 to the present time, some authors believe that ontology design methodologies are dependent the domain they are used for, while others hold the opinion that there could be a general all-purpose methodology usable in all domains. Also, regarding the description of the methodologies, it could been seen that some methodologies start from an abstract high-level and move downward through the details, while others start from raw-level information to reach the high-level concepts. Even the iterative methodologies that contain design cycles fall into one of these two categories depending on whether they start from abstract concepts or raw data.

The first attempt to propose an ontology design methodology traces back to the Cyc project started in 1984 and focused on developing a large-scale ontology to model the whole world [85–87].

After a few years, Grüninger *et al.* proposed a guideline for ontology design and a framework to evaluate the generated ontologies named Toronto Virtual Enterprise (TOVE) [88]. Another work presented in the same workshop TOVE was presented belongs to Uschold *et al.* that was a simple and flexible methodology emphasizing on the importance of documentation for sharing purposes [89]. They later published an improved version of their work [90].

Kactus [91] and Sensus [92] were other methodologies emerged in 1996. Kactus was based on reusing, integrating and refining the ontologies and Sensus provided a large-scale skeletal ontology containing more than 50,000 concept.

At this time, different ontology design methodologies emerged among which, METHONTOLOGY [93] could be counted as the most famous one. This methodology is developed based on the Institute of Electrical and Electronics Engineers (IEEE) standard for the development of a software life-cycle process [94].

Unified Process for ONtology (UPON) is another well-known ontology design methodology based on Unified Process (UP) and Unified Modeling Language (UML) defining design cycles. Each cycle includes four phases: inception, elaboration, construction and transition [95]. Each phase itself includes different iterations and each iteration contains five workflows: requirements, analysis, design, implementation and test. The authors of UPON validated their method using 18 evaluation metrics and compared their work against four other methods.

Another comprehensive ontology design methodology is NeON which is a “methodology for building collaboratively ontology networks” focusing on ontology requirements, scheduling and ontology reusing [96]. It supports nine different situations based on the available information for designing the ontology. NeON believes there should not be only one lifecycle for designing ontologies, as the software engineering world.

A quick and iterative ontology design methodology named Simplified Agile Methodology for Ontology Development (SAMOD) is proposed by Peroni [97]. This methodology consists of three iterative steps: “collect requirements and develop a modelete[1]”, “merge the modelete with current final model” and “refactor the current final model”. The whole process is described in details and enriched with tips to achieve the best results. The author evaluated his methodology from usability and learnability points of view with nine users.

As in this research we are going to model part of a cyber security system, finding a methodology designed for this context would be outstanding. Malenfant proposed an ontology design methodology named ATOM in the context of an expert security system [21]. His methodology aims to translate raw information into abstract concepts and leads to create an ontology in six main steps shown in Figure 3.1.

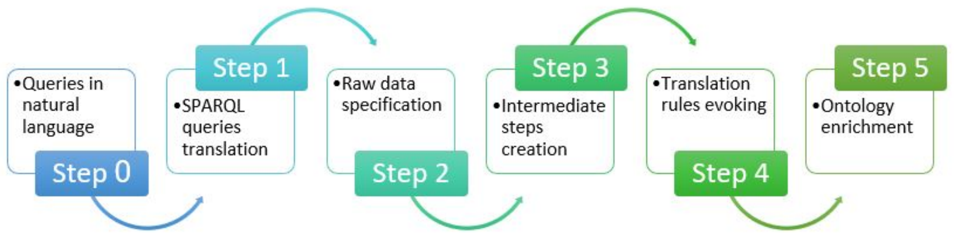

•Queries in natural language
Step 0
Step 1
•SPARQL queries translation
•Raw data specification
Step 2
Step 3
•Intermediate steps creation
•Translation rules evoking
Step 4
Step 5
•Ontology enrichment

Figure 3.1 Steps of ATOM

For evaluation, he considered five requirements that an ideal methodology has to address:

[1]A small stand-alone model focusing on a specific part of the domain used to prepare draft versions of the main model is called “modelete”.

1. Produce clear specification documents.

2. Avoid wasting too much time on modeling.

3. Explain step by step the process to be accomplished.

4. Produce high quality ontology (evaluated by OOPS [98] that could be shared and reused in other contexts.

5. The entire process had to be of reasonable duration for its application in industry.

ATOM is demonstrated to be able to address three of the above mentioned requirements (first, second and fourth).

ATOM assumes that the primary job of the security expert is to recover the multiple data generated by a system, then to classify, simplify and connect data in order to achieve the target. The most important achievement of ATOM is its data transformation which allows the analysts to move from a very crude representation of information to a more conducive and higher-level representation.

3.2.2 Ontology in insider IoC

Since there are many works suggesting an ontology for cyber security purposes, in this section, we narrowed down the search area and review the previous works focusing on using ontologies in insider IoC modeling and insider threat detection and prevention.

One of the first attempts to use ontology for detecting insider threats belongs to Aleman-Meza *et al.* who suggested using ontology to detect insiders unauthorized document access [99]. They proposed a method to extract the relationships between entities that indicate the reasons one tries to access a document and whether the document is relevant to the context of investigation or not. They did not evaluate their method completely, however, they reached reassuring early results.

Raskin *et al.* proposed using ontology to find unintended data leakage by processing employees' verbal and written communications such as blogs, Facebook, taped conversations, etc. [100]. The ontology was able to extract the semantic behind the language and detect contradictions or suspicious data leakage. Although such automated data leakage identifier allows the analysts to detect insiders more quickly, there is always the risk of spying on the employees. Also, many incomplete parts of the project makes its real practical results unclear.

An ontology for insider IoCs was proposed by Costa *et al.* to connect the insider threat descriptions and IoCs used by analysts for insider threat detection [101]. This ontology is built based on the natural language explanation of the cases reported by CERT® Insider Threat Center and is enriched by Structured Threat Information Exchange (STIX) [102] and Cyber Observable Expression (CybOX)[2] [103]. They also populated a small subset of their ontology using Windows system event logs that were translated to CybOX format. Two years later, the authors published a technical report of a knowledge base focusing deeply on insider threat technical indicators [13]. Although they have modeled a small part of behavioral and organizational indicators, the Software Engineering Institute (SEI) CERT framework could be counted as one of the most comprehensive frameworks developed for technical indicators.

The ontology developed by SEI CERT was extended by Greitzer *et al.* to support not only technical IoCs but also behavioral and organizational ones [104]. Their work focuses on insiders instead of events and contains sociotechnical factors. Later, the authors published another work named Sociotechnical and Organizational Factors for Insider Threat (SOFIT) which contains more than 300 indicators [105]. They demonstrated the importance of behavioral indicators in insider threat detection and prevention. In one of their latest papers published in 2019, they extended the work they had done on SOFIT which resulted in a more comprehensive ontology in insider threat domain [62]. They defined three main use cases describing different application types: sharing information, improving insider threat assessment and supporting insider threat scores. They also tried to visualize the results of SPARQL queries to have a more user-friendly interface. The ontology developed by SEI CERT was extended by Greitzer *et al.* to support not only technical IoCs but also behavioral and organizational ones [104]. Their work focuses on insiders instead of events and contains sociotechnical factors. Later, the authors published another work named SOFIT which contains more than 300 indicators [105].

As the aim of this research is to develop an ontology for insider threats within a financial institution, we were eager to find the works done in this environment and could find an ontology specifically designed for banking domain which is developed by Kul *et al.* [14]. Looking at the concepts defined for banking domain, one finds most of them related to the physical location of different branches. Also, the solution is not validated in a real financial environment.

An ontological framework for detecting insider threats including physical activities is proposed under the name of PS0 [106]. This framework can receive organization security policies and detect the users who are violating them with the help of a rule-based anomaly detection

[2]Currently, CybOX is integrated into STIX 2.0.

solution. Looking at a sample of rules mentioned in the paper, they conspicuously rely on employees arrivals and departures time. A few other rules try to find the behavior patterns violating security policies. The authors evaluated their method using 10 different use cases and assumed the analysts are familiar to query the system using SPARQL which is not always the case.

3.3 Justification of the methodology

Since one of the main challenges in this research is having different levels of granularity in data sources, we needed an ontology design methodology that addresses the problem of semantic gaps between abstraction levels. After reviewing the proposed ontology design methodologies, we found ATOM meeting this criteria. Also, to the best of our knowledge, among all design methodologies proposed, only ATOM is developed specifically to be used in cyber-security environments. Therefore, we decided to follow this methodology as insider threat detection is an important part of a cyber-security system. Moreover, we were eager to evaluate how realistic is to use ATOM in a real cyber security environment.

Among all different frameworks and tools, in this research we chose Protégé for designing the ontological model as it is a free software with a user-friendly User Interface (UI). Protégé allows for visualizing classes and their relationships and is capable of executing SWRL, DL and SPARQL queries. The hierarchical schema of the concepts provided by Protégé makes it a great tool for explaining the details to the people who are not familiar with semantic web concepts. This makes extending and modifying the ontology easier.

For populating and querying the database, we chose GraphDB since it is a very powerful and light-weight free software. According to the fact that the main codes of the project are written in Python to be compatible with the infrastructures of the target network, GraphDB was one of the best options as its Application Programming Interface (API) could be used by python libraries simply and effectively.

3.4 Summary

In this chapter we first summarized the basic knowledge of ontology. Since there are many concepts and languages in ontology and semantic web domain, we decided to briefly introduce the ones that are used in this project. Moreover, some of the most well known software and tools developed to design, store, query and make inferences are mentioned.

In the second section we aimed to review the works done in two main areas: First, we mentioned some of the most well known ontology design methodologies. Then, we summarized the state of the art in ontology usage in insider threat demonstrated the effectiveness of using ontologies in this area. However, we could not find a research on inisder kill chain detection using ontologies.

Finally, we justified the selection of ontology design method and the tools for designing and communicating with the ontological database in section 3.

Another lesson learned from the literature is that not all the security analysts are willing to learn a new query language and ontology concepts. Therefore, the best solution is to benefit from ontology strength in the background and prepare a UI that analysts are comfortable working with.

We are going to design an ontology to model insider threat indicators in Desjardins, find kill chain patterns and visualize the result so that not only the analysts do not need to be familiar with specific software or query languages but also they could take advantages of ontological model while the visualization process decreases the investigation process time.

CHAPTER 4 RESEARCH ENVIRONMENT AND DATA

To acquire the required knowledge for designing the ontology, we need to get familiar with all the concepts defined and used in Desjardins insider threat investigation environment. In this chapter we take a look at a summary of the environment specification and review the data we had access to.

4.1 Environment

Desjardins insider threat investigation environment specification and procedures are summarized in this section.

4.1.1 Human resources overview

Currently, insider threats are investigated by eight analysts in a team named "*Équipe investigations de sécurité de l'information*". The analysts monitor Around 48,000 users as well as 5,000 consultants in one shift. The detection process is mostly rule-based, however, there are ongoing projects involving anomaly detection and predictive analysis. There are around 700 indicators defined coming from 10-20 data sources that are fitted into a standard format.

4.1.2 Insider indicator collection procedure

There are diverse security devices and applications in Desjardin's network. Different security policies are written for each of them to generate an IoC when there is an insider suspicious activity. For example, when a user tries to attach a big file to an email, a security policy triggers an indicator event. All of these indicator events with different formats coming from a variety of sources will be collected in a center and will be translated to a unique list of IoCs known for the system (Figure 4.1).

4.1.3 User interface

The product that is used to create reports from received indicator events is Microsoft PowerBI [107] which is an enterprise Business Intelligence (BI) platform that is connected to data and visualizes it to provide deeper insights. It supplies various reports such as:

- Employees departures (*e.g.* top evolution of the risk rating since the announcement of an individual's departure)

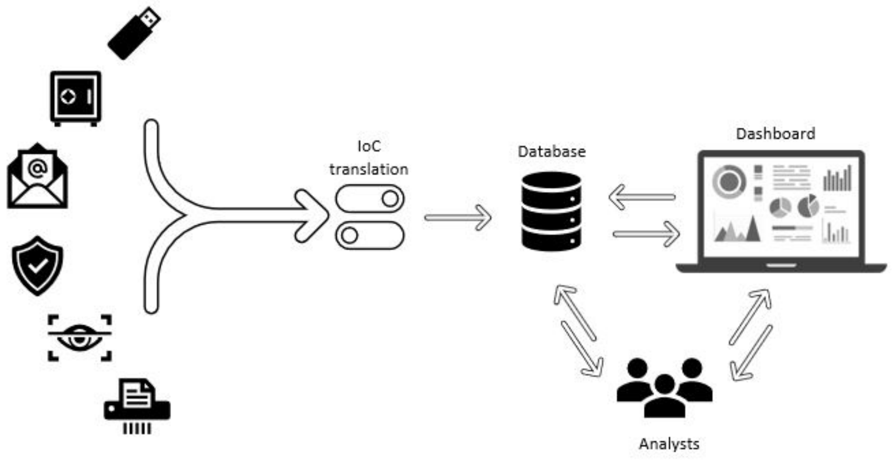


Figure 4.1 Desjardins insider IoC investigation environment

- Hunts (*e.g.* top risk ratings for exfiltration)
- Operations summary (*e.g.* number of ongoing/open/closed investigations)
- Metrics (*e.g.* detailed list of recent events)
- Organization (*e.g.* user's info and what he has done recently in terms of risk factor, kill chain steps taken and so on)
- Profile sheet (*e.g.* risk rating per category, user, etc.)
- Monitoring summary (*e.g.* received IoCs, list of tasks, etc.

4.1.4 Insider indicator analysis procedure

Using the reports and dashboards generated in PowerBI platform, the analysts could see what users look suspicious and what they have done. To do further investigations, analysts might need to query the database or examine other security tools to acquire information which is not provided by the user interface. Based on the result of this analysis, they take the appropriate action.

4.2 Data

In this research we were provided sample data files as well as some basic configuration data that are described in the following.

4.2.1 Indicators

According to the latest reports observed, Desjardins has defined 725 IoCs in which 94.06% are single indicators and the rest are use cases. A single indicator shows an activity or a status and is generated by one security policy or device. However, a use case is a combination of indicators happening simultaneously. For example, a use case of a high access account is reported if the user generates specific privilege indicators (such as having the right to access USB port, having high privilege account, etc.) in the same day.

According to the source of indicator event and its nature, indicators are categorized in 24 groups (Figure 4.2). Use cases are categorized in two main groups: Access (32.55%) and Leak (67.44%). The format they used for the indicators and use cases can be mentioned as:

$$CC - [source\_name] - ddd - dd \tag{4.1}$$

where CC is equal to “IN” or “UC” for indicators and use cases respectively. “ddd” and “dd” are padded numbers showing the message type. The second number usually indicates a variant of the first number.

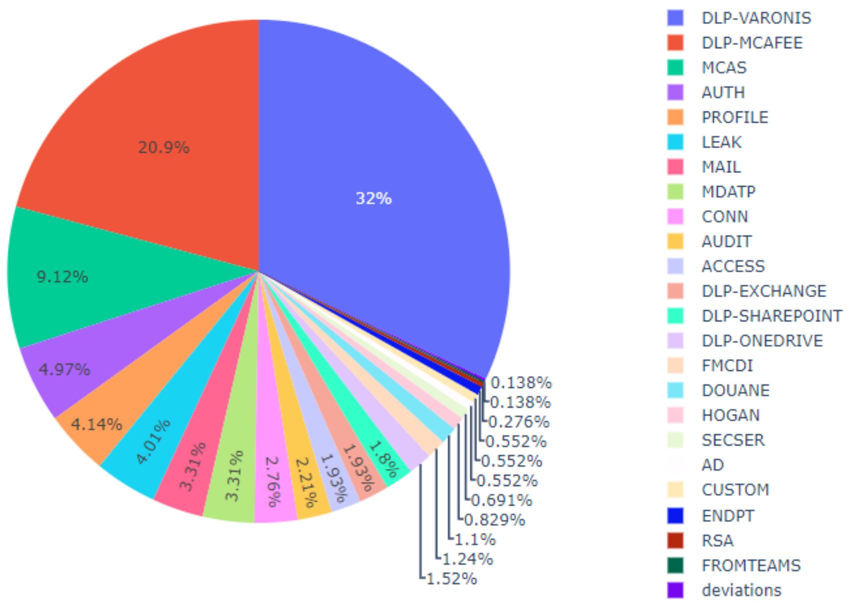


Figure 4.2 Distribution of indicator event sources

Moreover, a risk factor is assigned to each indicator, based on its importance and sensitivity. Each user also has a risk factor. Whenever a user generates an indicator, his/her risk factor sums up with the risk factor of the newly generated indicator. For example, if a user generates one indicator with the risk value of 100 and two indicators with the risk value of 10, his/her risk factor will be 120.

4.2.2 Kill chain

Desjardins has designed its own insider threat kill chain to detect insider data leakage. The kill chain consists of 5 main steps: Reconnaissance, Privilege, Accumulation, Execution and Cleanup all described in Table 4.1.

Table 4.1 Desjardins insider threat kill chain

Step	Description
Reconnaissance	User navigates in the shared directories or tries to access the database, confidential data, etc.
Privilege	User might need to escalate his/her privilege or steal someone else's one.
Accumulation	It takes some time to gather all the required data. It can't usually been done at one step.
Execution	User tries to exfiltrate data. Sometimes he/she tests the defense system before a complete exfiltration by starting to exfiltrate a small sample of data to see what will happen. Then he/she finds a safe channel to export all data. This step can be done using different channels like uploading data somewhere, using USB drives, sharing or printing, etc.
Cleanup	Finally, User tries to remove all the traces, logs, data, etc that are related to the threat.

The aforementioned IoCs have an attribute named "category" demonstrating which step of the kill chain they belong to. The frequency of generating indicators categories is shown in Figure 4.3. Since there are indicators that are not categorized yet, there is an extra category named "other".

4.2.3 Indicator event

According to the sample log files, each indicator event appears in the format below:

$$UserID, Period(daily, weekly, etc.), Date, IndicatorID, MetricValue, lastUpdate$$

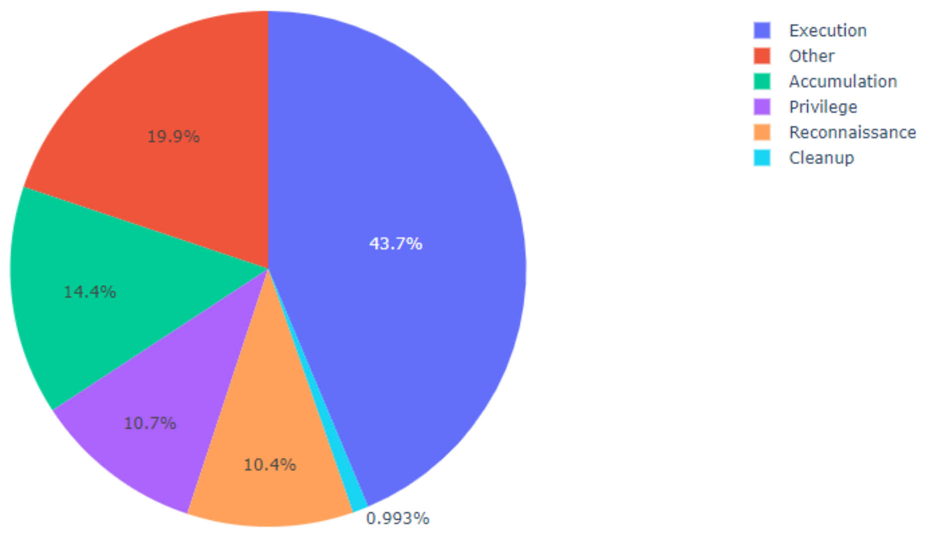


Figure 4.3 Distribution of indicators categories

We could not see any indicator event happening during night. The MetricValue parameter gives extra information about some special indicators. For example, for the indicators showing a user takes screenshots, MetricValue contains the number of screenshots taken.

4.2.4 Statistics and data analysis

During the observation phase, we used a total number of 2,003,166 indicator events that happened in 461 distinct dates (from 2020-01-05 to 2021-04-09) generated by 453 different users. We observed that visualizing the total number of indicator events per date could help with detecting peaks in number of indicator events, which can be an important event to be investigated. As shown in Figure 4.4 the peaks are quickly detectable.

The histogram of number of indicator events per date asserts most of the time, we have between 3000 – 5500 indicator events every day (Figure 4.5). It can be inferred that the normal interval for frequency of indicator events per date is [3000,5500]. In a very simple approach, all the dates with total number of events outside of this interval could be counted as an anomaly and should be investigated. However, defining the normal number of events per date requires more detailed information.

Another useful plot would be the plot of kill chain steps or category of indicators per date. As it is shown in Figure 4.6 privilege and Reconnaissance are the first and second most occurred kill chain steps. The other three steps are happening rarely in comparison with these two.

As it is mentioned before, each user has an attribute named "risk factor" demonstrating how

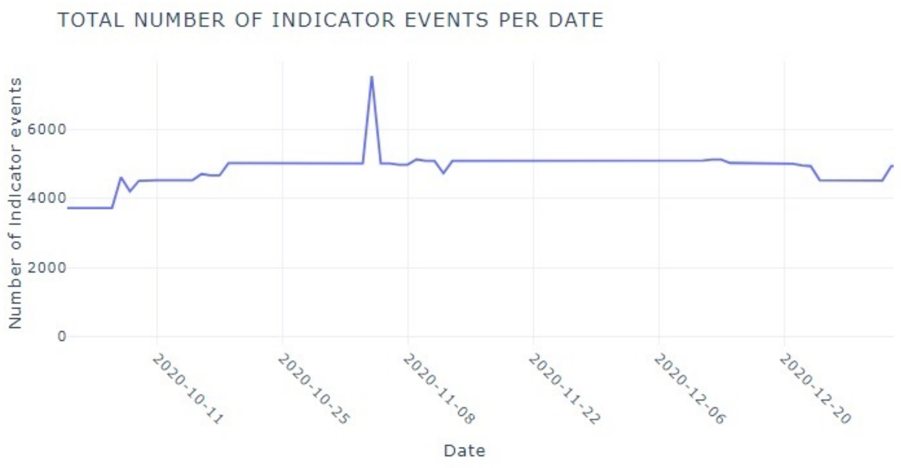


Figure 4.4 Total number of indicator events per date

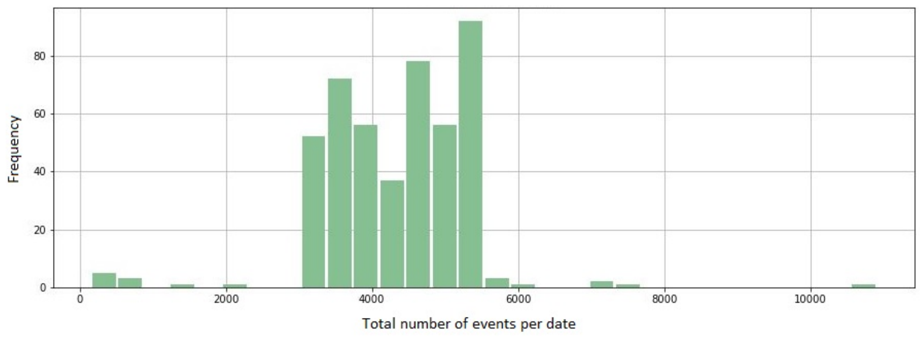


Figure 4.5 Histogram of number of indicator events per date

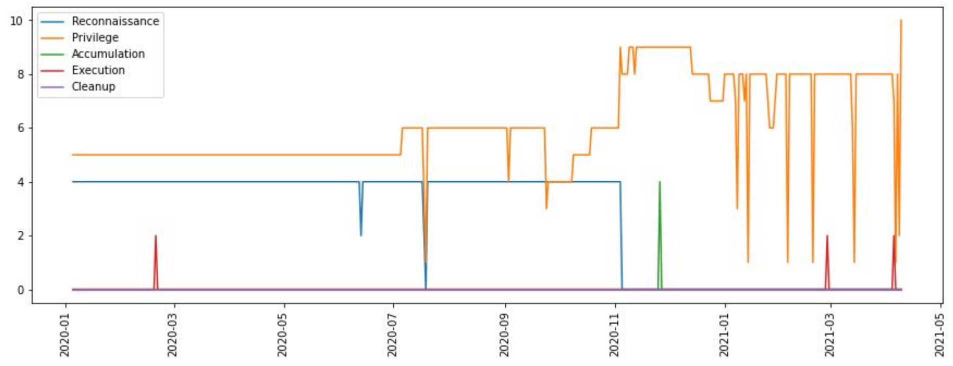


Figure 4.6 Total number of kill chain steps taken per date

many suspicious IoCs the user has generated. This value is the sum of risk values related to the indicators each user generates in a specified time window. According to the observed logs, top 10 users (with anonymized usernames) who have the highest risk factor are presented in Table 4.2. These values could provide us hints about the numbers we are dealing with.

Table 4.2 Top 10 users with highest risk factor

Username (anonymized)	Risk factor
2ef3069055430c7fe858bc572ec59c07dc5de2e8345b2631fd5c2eb9e61fb07b	4778
d23f842dda0c77cd83f5f60d6f65d673c8125fc513b345f433b49c43b7f17c00	4629
2c8eefed34f9b3beef2b4f7c8bf537a68141695872b3f9560744032a21e023f4	4265
01dc071cb7b438851b70d325dc6ee09ae2b00cc460026daaa3ca54565196f83d	3890
fe90490fc7aae1ae0e8718d4de4f0cdce57668b37da27e46b25d78d14249b6a1	3836
48ae015e61d67b3abb9ab0aa2d068d6ce60d194dc1f76e87d9ff3d9f211a1634	3274
1523ed1395ce63753a192a49646e892af63eea95532c6a4fa0757bf1a78a3dc8	3236
76d57ab15bce771e737469bb6a0858e2de8fba2df34eb77437ee81983910d834	3058
e545b683c4a8302bd43d93790561b68a91d4cf131e33beb9b7957f14a3306fc2	2702
6e4f406b26effab153584b9567663e92b93b2bbaf49306ca6a4bdedcc5686f5b	2592

The risk factors are calculated during the whole observation time window which lasts more than a year.

4.3 Summary

In this section the environment specification and analysis of the data we had access to were presented and the main concepts and relationships required for designing the ontology are identified.

The analysts can investigate the insider threats using current dashboard and query the database to acquire more knowledge about a specific user or date. As monitoring this huge amount of data is very time consuming, we aim to automate this process by providing an interactive visualisation tool so that instead of querying the database, they could click on the data they need and see the results. Also, we would like to analyze each user inside its context as some literature propose to consider team work roles to decrease false positive rates. Modeling users relationship is a use case that is better to be done by ontology. Otherwise, we will have redundancies in relational databases. Finally, by focusing on kill chain activity, our approach is able to recognize threats in their initial stages.

CHAPTER 5 INTERACTIVE VISUALIZATION OF INSIDER INDICATOR ONTOLOGY (IVIIO)

In this chapter, the methodology and details of our solution named IVIIO are described in three main parts: system architecture, ontological solution and visualization. We first go through the system architecture. Ontology design process and population are explained in next section and the details of visualization are provided in the last section.

5.1 System architecture

The abstract model of our solution is shown in Figure 5.1. As demonstrated before, we do not have access to the data generated by sensors nor their input traffic. Data will be digested by the populating module after being normalized and will be used to populate the ontological model. Since the values of some of data properties such as "latestStepReached" and "latestStepReachedDate" should be updated during the population module, there are two vertical arrows between population module and ontological model. Whenever the user is proceeding in kill chain, we need to query the knowledge base, get the values of related properties and update them. Finally, the dashboard will connect to the knowledge base to visualize the required data so that the analysts do not need to learn how to directly work on ontological database. Also, they cannot modify the fundamental information such as indicator details, users data, kill chain definition and so on.

Figure 5.1 clearly defines two main parts of the solution as: ontological solution and visualization. In the following, we will describe each part and explain the methodology used for design and implementation.

5.2 Ontological solution

In this section, we explain the process of developing an ontological solution from the first step of designing ontology to making an ontological model ready to use.

5.2.1 Design

In chapter 3, we reviewed a variety of ontology design methodologies and decided to use ATOM. In the following, this ontology design process is explained in the format of different steps demonstrated in Figure 5.2.

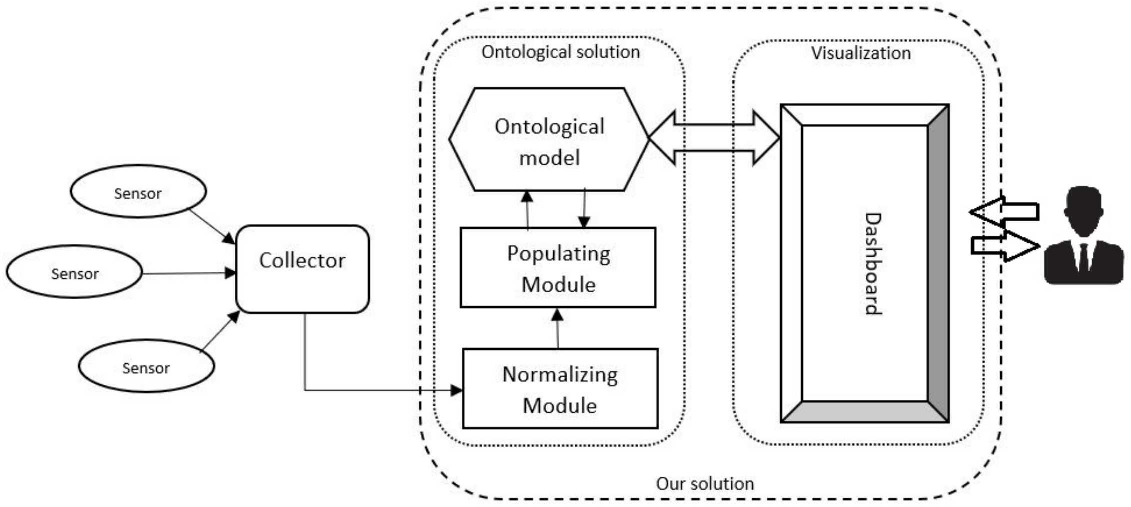


Figure 5.1 IVIIO architecture

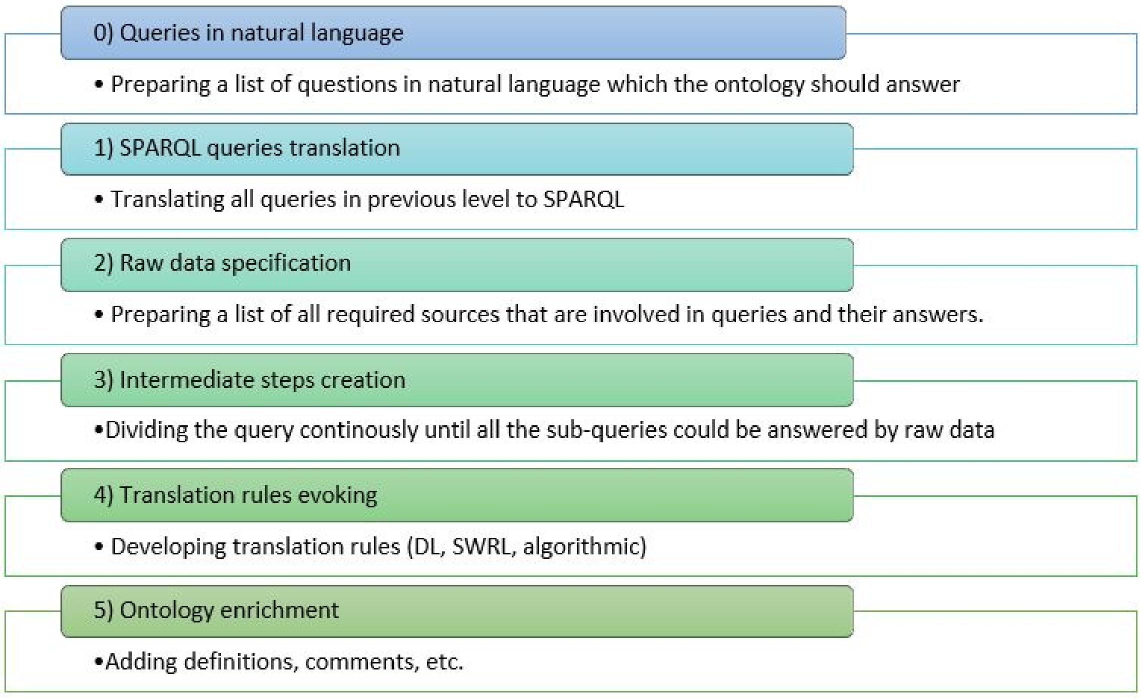

0) Queries in natural language
• Preparing a list of questions in natural language which the ontology should answer
1) SPARQL queries translation
• Translating all queries in previous level to SPARQL
2) Raw data specification
• Preparing a list of all required sources that are involved in queries and their answers.
3) Intermediate steps creation
•Dividing the query continously until all the sub-queries could be answered by raw data
4) Translation rules evoking
• Developing translation rules (DL, SWRL, algorithmic)
5) Ontology enrichment
•Adding definitions, comments, etc.

Figure 5.2 Description of ATOM steps

Phase 0: Queries in natural language

Based on ATOM methodology, at first, we need to identify the questions to be responded by the ontology. For this purpose, we prepared a list of necessary reports inspired by studying the currently used dashboard and investigation process. Also, according to the feedback from experts in Desjardins, we added some new reports to enhance the investigation process by focusing on kill chain detection. The final questions ontology must be able to answer are listed the following ones:

1. Who are the top 10 users with the highest risk factor?
   Each user has a risk factor which is calculated based on the risk factors of his/her generated IoCs. Reporting the users who have the highest risk factor helps to identify suspicious users.

2. What are the top 10 most generated indicators?
   Having the most generated indicators helps to indicator refining while being useful for user behavior analysis purposes.

3. Which steps of the kill chain are taken the most?
   We need to see how frequently different steps of the kill chain are taken to refine IoC categorization and get the big picture of the nature of the generated indicators.

4. What is the pattern of kill chain steps taken by a specific user in comparison with his/her teammates?
   For investigation purposes, it is useful to look at the user behavior in terms of different steps of the kill chain taken in different dates. In some cases, we need to compare the number of kill chain steps taken by a specific user with the team average.

5. Who have reached to the highest step of the kill chain?
   Since high risk users are not always malicious, we need to find suspicious users based on another point of view. This query lists the users who have reached to the highest step of the kill chain.

These questions are verified by experts in Desjardins to fulfill their requirements.

Phase 1: SPARQL queries translation
In this phase the questions defined in the previous phase are translated to SPARQL queries demonstrated in Table 5.1. In order to provide the filters required by the analysts, we consider date stamp filters (as *"start_date"* and *"end_date"*) and username filter (as *"username"*) in SPARQL Queries. In the following, each query is explained:

1. Q1: To calculate top 10 high risk users in a specified time window D1, we need to query all the events happening in D1 and grouping them based on the username of the user who generated the event. Since each event has an indicator and each indicator has a risk factor, we can sum up the risk factors of the indicators related to the events

generated by each user (named as totalRisk). Finally we report the usernames and their total risk sorted high to low and limited to 10 result to see top 10 users with highest risk factor.

2. Q2: In order to have a list of top 10 most generated indicators in a specified time window D1, we query all the events happening in D1 and grouping them based on their indicator type. Finally we report the indicator ID and its frequency for all of the indicators related to the events happening in D1. The result is sorted based on descending frequency and is limited to 10 outputs.

3. Q3: For listing steps of the kill chain taken by the users, we do the same process as Q2, however, this time we group the results based on kill chain step of the indicator related to each event (we assume each indicator is mapped to a kill chain step).

4. Q4: In this query we need to compare user activity versus the team average. Therefore, we need two queries: Q4.1 for a single user activity and Q4.2 for the team average.

   Q4.1: To prepare a list of kill chain steps taken by a user in different dates, we query all of the events generated by this specific user who has a username shown as *"username"*. The result is grouped based on date and kill chain step taken. Finally, we report the date of the event, kill chain step taken and frequency of the events.

   Q4.2: In this query we need to report the average number of different kill chain steps taken by the teammates of *"username"* in different dates. To do so, first, we find all of the users who work in the same team as *"username"* except himself/herself. Then we do the same process as Q4.1 to report the dates, kill chain steps and frequency of the events. This time we also report username since there may be more than one teammate. Finally, we report the average of the frequencies related to the same dates and same kill chain steps.

5. Q5: To report the users who have reached to the latest step of the kill chain, we assume for each user, there is a property named :latestStepReached which shows the latest step of the kill chain that user has reached. Therefore, at this level we only need to query the users and their latest step reached and sort the result based on descent step number. We will go through the details of :latestStepReached property in the next phases of methodology.

Table 5.1: SPARQL queries

ID	**SPARQL queries**

Q1	PREFIX : <http://www.ontology.ca/insiderontology-3.0#>
	SELECT ?username (SUM(?r) AS ?totalRisk) WHERE {
	  ?u a :User;
	    :hasUserName ?username.
	  ?e a :IndicatorEvent;
	    :hasUser ?u;
	    :hasEventDate ?date;
	    :hasIndicator ?i.
	  ?i a :Indicator;
	    :hasRisk ?r;
	  FILTER((?date>=“*start_date*”) && (?date<=“*end_date*”))
	}
	GROUP BY ?username
	ORDER BY DESC(?totalRisk)
	LIMIT 10

Q2	PREFIX : <http://www.ontology.ca/insiderontology-3.0#>
	SELECT ?indicator_id (COUNT(?indicator_id) AS ?frequency) WHERE {
	  ?event a :IndicatorEvent;
	    :hasEventDate ?date;
	    :hasIndicator ?ind.
	  ?ind a :Indicator;
	    :hasIndID ?indicator_id.
	  FILTER((?date>=“*start_date*”) && (?date<=“*end_date*”))
	}
	GROUP BY ?indicator_id
	ORDER BY DESC(?frequency)
	LIMIT 10

Continued on next page

Table 5.1 – continued from previous page

ID	SPARQL queries

Q3	PREFIX : <http://www.ontology.ca/insiderontology-3.0#>
	SELECT ?stepName (COUNT(?stepName) AS ?frequency) WHERE {
	  ?event a :Event;
	    :hasIndicator ?ind.
	  ?ind a :Indicator;
	    :hasKillChainStep ?step.
	  ?step :hasStepName ?stepName
	  FILTER((?date>=*“start_date”*) && (?date<=*“end_date”*))
	}
	GROUP BY ?stepName
	ORDER BY DESC(?frequency)
	LIMIT 10

Q4	Q4.1 and Q4.2

Q5	PREFIX : <http://www.ontology.ca/insiderontology-3.0#>
	SELECT ?username ?step WHERE {
	  ?user a :User;
	    :hasUserName ?username;
	    :latestStepReached ?step;
	  ?step a :KillChainStep. }
	ORDER BY DESC (?step)
	LIMIT 10

As mentioned in Table 5.1, fourth query contains two different queries mentioned in Table 5.2 that are supposed to be compared together.

Phase 2: Raw data specification

For each query determined in the previous step, we should extract the raw information needed.

Query No.1: There are four classes in this query that need to be instantiated: :User, :Sensor[1], :Indicator and :IndicatorEvent. Therefore, as ATOM proposes, we define four drivers described in Table 5.3. Due to the fact that there are over 700 distinct indicators, all

[1]This class is defined based on provided data but is not used in this version of the project.

Table 5.2 Subqueries for Q4

ID	**SPARQL Query**

Q4.1	PREFIX : <http://www.ontology.ca/insiderontology-3.0#>
	SELECT ?date ?stepName (COUNT(?stepName) AS ?frequency) WHERE {
	  ?event a :IndicatorEvent;
	    :hasUser ?user;
	    :hasEventDate ?date;
	    :hasIndicator ?ind.
	  ?user a :User;
	    :hasUserName *"username"*.
	  ?ind a :Indicator;
	    :hasKillChainStep a [:KillChainStep;
	      :hasStepName ?stepName].
	}
	GROUP BY ?date ?stepName

Q4.2	PREFIX : <http://www.ontology.ca/insiderontology-3.0#>
	SELECT ?date ?stepName (AVG(?c) AS ?average_frequency) WHERE{
	  SELECT ?user ?stepName ?date (COUNT(?e) AS ?c) WHERE {
	    ?u a :User;
	      :hasUserName *"username"*;
	      :worksIn ?team.
	    ?user :hasUserName ?un;
	      :worksIn ?team.
	    FILTER(?un!=*"username"*)
	    ?e a :IndicatorEvent;
	      :hasEventDate ?date;
	      :hasUser ?user;
	      :hasIndicator ?i.
	    ?i a :Indicator;
	      :hasKillChainStep [a :KillChainStep;
	        :hasStepName ?stepName].
	  }
	  GROUP BY ?user ?date ?stepName
	}
	GROUP BY ?date ?stepName

sharing the same properties, instead of creating a subclass for each indicator, we decided to define them as instances of the class :Indicator.

Query No.2: All the classes that are used in this query are considered in Q1.

Query No.3: In this query, we notice there is an extra property for the class :Indicator named :hasKillChainStep which is linked to a new class. We call this class :KillChainStep and we need its instances to have two properties[2] shown in Table 5.4.

Query No.4: Considering both queries required to answer to the fourth question, we noticed that there are new elements in Q4.2 to be added to the ontology. class :User needs a new property named :worksIn which connects the class :User to a new class named :Team. This new information is summarized in Table 5.5.

Query No.5: In this query, a new property for class :User named :latestStepReached is identified (Table 5.6). The configuration files containing indicators specification and kill chain steps are used by drivers to create required instances.

Phase 3: Intermediate steps creation

As the exact information about the data sources and employees is highly confidential, we only examine the high level indicator events received in the central collector. In this case, raw data that we have access is actually a high level data acquired by internal processes. For this reason, we do not have much intermediate levels.

The intermediate level developed in this phase distinguishes behavioral and non-behavioral IoCs where required. Due to the fact that the analysts were more willing to investigate what the user has done, the indicators showing activities matter more. Therefore, we decided to categorize all of the indicators in two main groups: behavioral and non-behavioral. Each indicator showing a user activity is labeled as behavioral and the rest which are related to other domains such as user profile are labeled as non-behavioral. For example, an indicator showing a user is copying files is considered to be behavioral while another indicator demonstrating the fact that a user does not have a high privilege account is considered to be non-behavioral. In order to provide the option to monitor different types of indicators (behavioral, non-behavioral or both), for each query (except Q5) we need to define whether the indicator event is related to a behavioral or non-behavioral IoC. This intermediate step (IQ1) shown in Figure 5.3 is not applicable for Q5 as the kill chain attacks contain both behavioral and non-behavioral IoCs. Therefore, we should consider both groups.

Phase 4: Translation rules evoking

[2]In this query we only need the property :hasStepName. However, another property named :hasStepNumber is defined according to the data provided in which, for each kill chain step, we have a name and a number.

Table 5.3 Drivers defined for Q1 and Q2

Driver name	**Driver description**	**Properties**
D001	Creates instances of the class :Sensor	:hasSensorID :hasSensorDesc
D002	Creates instances of the class :User	:hasUserName
D003	Creates instances of the class :Indicator	:hasIndicatorName :hasIndID :hasRisk
D004	Creates instances of the class :IndicatorEvent	:hasIndicator :hasUser :hasEventDate

Table 5.4 Drivers defined or modified for Q3

Driver name	**Driver description**	**Properties**
D003	Creates instances of the class :Indicator	:hasIndicatorName :hasIndID :hasRisk :hasKillChainStep
D005	Creates instances of the class :KillChainStep	:hasStepName :hasStepNumber

Table 5.5 Driver modified for Q4

Driver name	**Driver description**	**Properties**
D002	Creates instances of the class :User	:hasUserName :worksIn

Table 5.6 Driver modified for Q5

Driver name	**Driver description**	**Properties**
D002	Creates instances of the class :User	:hasUserName :worksIn :latestStepReached

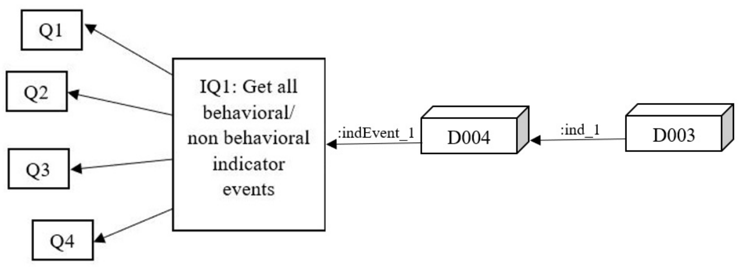


Figure 5.3 Intermediate query IQ1

To answer query Q5, we need to calculate the latest step of the kill chain each user has reached and then, try to sort the users who have reached the highest step of the kill chain. As shown in Figure 5.4 we need to find a translation rule to connect raw data to the query. Therefore, we deployed two approaches that are explained in the following.

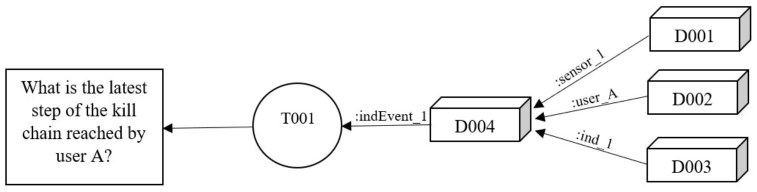

What is the latest step of the kill chain reached by user A?
T001
:indEvent_1
D004
:sensor_1
:user_A
:ind_1
D001
D002
D003

Figure 5.4 Addition of translation rule T001

To store the latest step each user reaches, we do the process inside the populating code. In this way, we only need to add another data property for the class :User named as :latest-StepReachedDate. Therefore, for each user we know what is the latest step of the kill chain he/she has taken (:latestStepReached) and when it has happened (:latestStepReachedDate). A simplified version of the code that is responsible to update the values of :latestStepReached and :latestStepReachedDate properties can be seen in Table 5.7.

As mentioned before, normalization module sorts the logs based on username first and then dates. Therefore, the log is separated to different sections, each related to a specific user.

Table 5.7 Translation rule T001

ID	Description

T001	if(currentStep == prevStep+1 and currentDate>=prevDate):
	prevStep+=1
	prevDate = currentDate

Considering one section, the population script will query the knowledge base to retrieve the values of :latestStepReached and :latestStepReachedDate properties of the current user and store them in prevStep and prevDate variables respectively. The currentStep and currentDate variables contain the related information in the newly received indicator event. If the new indicator event specifies the user is taking the next kill chain step, the values of :latestStepReached and :latestStepReachedDate properties will be updated. In order to minimize the number of queries, we will update these values at the end of each section where there is no more information about the current user.

Phase 5: Ontology enrichment
In this phase, we tried to enrich the ontology using comments and labels. We also considered some information that are not used in our queries but might be useful for future works. In the following we list these additional entities.

1. Classes

   Five classes developed by previous students in the lab were reused to make the ontological model integratable with other models defined for different domains. These classes are :Attack, :Context, :Event, :Vulnerability and :Machine. Class :Sensor is also added to the model to represent the sensor information provided by Desjardins. We tried to model as much entities in real environment as possible to be used in future works.

2. Properties

   According to new classes added to the ontology, we needed to define properties to model the new relationships. Therefore, new properties such as :causes, :happensIn, etc. are appended.

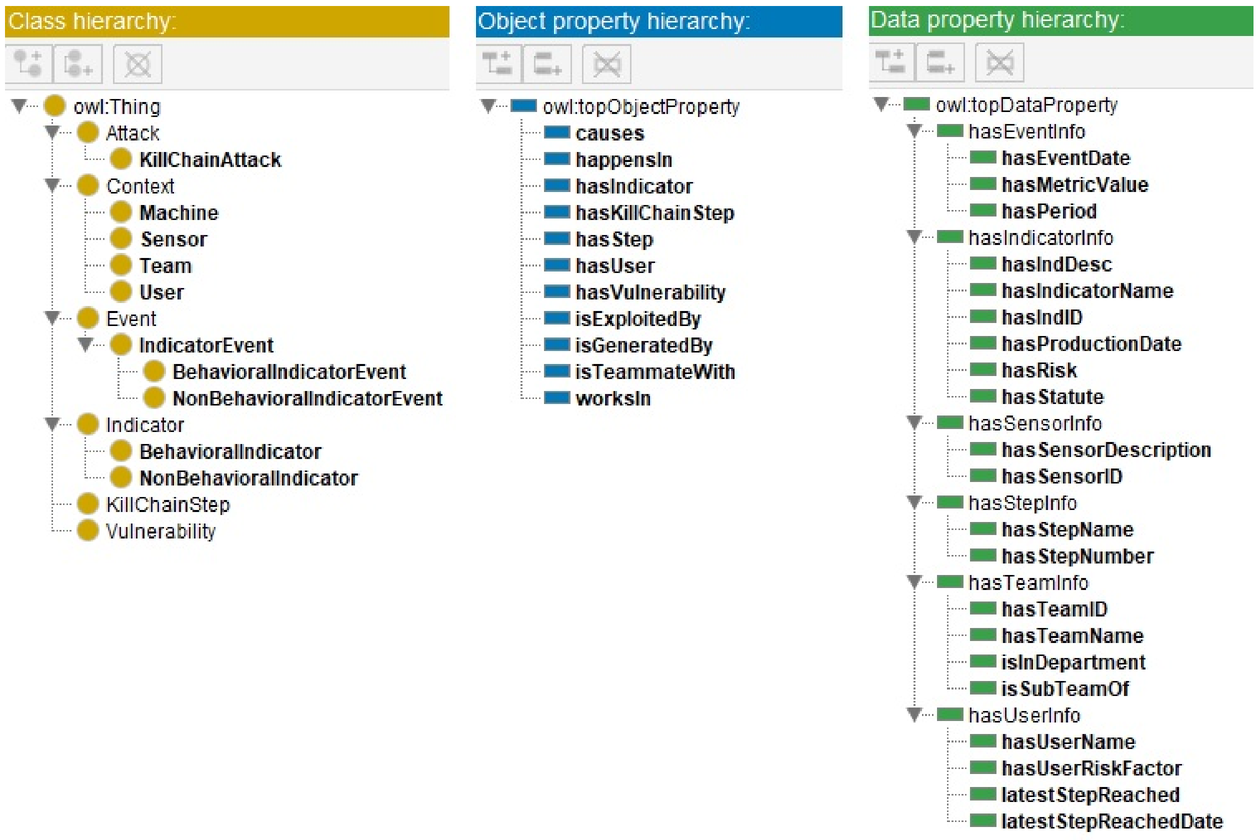
3. Comments

   In order to increase human readability, we have added labels and comments to the classes and properties. Thus, modifying and extending the ontological model would be facilitated.

The final schema of all the defined classes and properties are summarized in Figure 5.5.

5.2.2 Implementation

The ontological model designed using Protégé will be imported in a GraphDB repository, so that we could query the ontology and populate it inside the code. We deployed a Python library to connect to GraphDB named SPARQLWRAPPER [108].



```
Class hierarchy:
owl:Thing
    Attack
        KillChainAttack
    Context
        Machine
        Sensor
        Team
        User
    Event
        IndicatorEvent
            BehavioralIndicatorEvent
            NonBehavioralIndicatorEvent
    Indicator
        BehavioralIndicator
        NonBehavioralIndicator
    KillChainStep
    Vulnerability
```

```
Object property hierarchy:
owl:topObjectProperty
    causes
    happensIn
    hasIndicator
    hasKillChainStep
    hasStep
    hasUser
    hasVulnerability
    isExploitedBy
    isGeneratedBy
    isTeammateWith
    worksIn
```

```
Data property hierarchy:
owl:topDataProperty
    hasEventInfo
        hasEventDate
        hasMetricValue
        hasPeriod
    hasIndicatorInfo
        hasIndDesc
        hasIndicatorName
        hasIndID
        hasProductionDate
        hasRisk
        hasStatute
    hasSensorInfo
        hasSensorDescription
        hasSensorID
    hasStepInfo
        hasStepName
        hasStepNumber
    hasTeamInfo
        hasTeamID
        hasTeamName
        isInDepartment
        isSubTeamOf
    hasUserInfo
        hasUserName
        hasUserRiskFactor
        latestStepReached
        latestStepReachedDate
```

Figure 5.5 Classes and properties of the ontological model

To populate the ontology using real data samples, we developed a two-stage population code. In the first stage, we remove the useless fields and sort the input data based on username and date to have the lowest possible number of queries during population phase. We will describe how this normalization module affects the population phase in Phase 4 of ontology design process. As it is mentioned in the design process, we decided to update the values of the latest step of the kill chain reached by every user inside the code. Therefore, we sort the data based on username and date subsequently, so that in the population phase, the value of :highestStepReached property for each user is updated only once if necessary.

We begin with populating the basic concepts (such as indicators, steps of the kill chain, etc.). Then, the indicator event records will be added to the ontological database. For each indicator event, we store all the related data for investigation purposes.

The overall schema of the designed ontology is shown in Figure 5.6. We reused the main concepts defined by previous works [20,22], namely Attack, Context, Event and Vulnerability. Other classes and relationships are extracted during the design process.

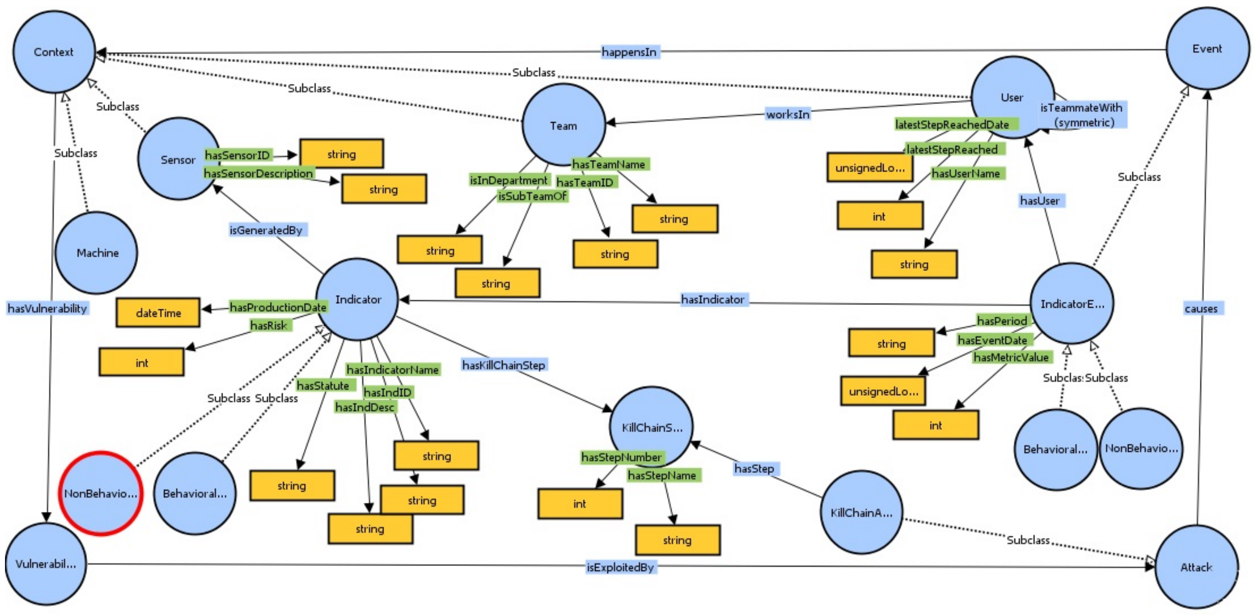


Figure 5.6 Designed ontology schema

5.3 Visualization

In this section, we will go through the design and implementation details related to the visualization module.

5.3.1 Design

Considering the main questions to be answered, we followed the procedure introduced in [7] shown in Figure 5.7. Each phase is described in the following.

Phase 1: Visualization goals

Since we already defined the main queries for the ontology, here we have the same questions as the visualization goals. Also, the required features such as being interactive is assumed as a goal. Table 5.8 lists the goals in the first process iteration.

One of the refinements done in the next iterations was related to VQ4. We noticed that a detailed view of user kill chain activity is required so that the analyst could find patterns by looking at the visualized report. Also, we needed to compare user activity versus his/her teammates. Therefore, we decided to divide the task to two main parts: VQ4.1: single user

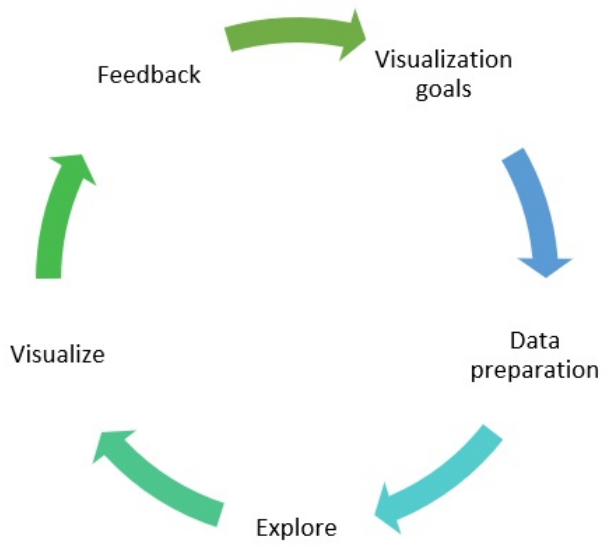


Figure 5.7 Security data visualization process [7]

analysis and VQ4.2: team analysis. In the first part, the kill chain activity of a user is visualized. Second part mostly focuses on comparing user activity versus the team average.

It also should be mentioned that during the visualization process, we figured out some changes that are needed for the SPARQL queries to work correctly. For example, Q4.2 calculates the average number of indicator events considering only active users. While there might be other teammates who are not generating any event but should be considered in the calculations. For example, in a team of five employees, there might be only two users generating indicator events. The average number of indicators generated by these two users cannot be reported as the team average since there are three other users generating no indicator. Small changes such as querying the number of teammates could solve these kind of problems.

Phase 2: Data Preparation

There was some missing data that was addressed in this phase. We tried to deal with this problem using two controls:

- Filling missing information: producing data where applicable.
- Data cleaning: removing the rest of missing information where it can make the reports confusing.

The indicators list we had access to, did not clearly separate behavioral indicators from non-behavioral ones. However, we could produce this missing information and determine

Table 5.8 Visualization goals in the first iteration

Index	Goal type	Goal description
VQ1	Question	Who are the top 10 users with the highest risk factor?
VQ2	Question	What are the top 10 most generated indicators?
VQ3	Question	Which step of the kill chain is taken the most?
VQ4	Question	What is the pattern of kill chain steps taken by a specific user in comparison with his/her teammates?
VQ5	Question	Who are the users who have taken the highest step of the kill chain
VF1	Feature	Date filter
VF2	Feature	Username filter
VF3	Feature	Interactive diagrams

whether the indicator is behavioral or non-behavioral with the help of description field in the indicators detailed list.

To exemplify a case in which data needed to be cleaned, we can mention invalid kill chain steps or risk factors assigned to the indicators (such as “other”, “to be determined”, blank, etc.). To address these problems, we tried to provide the correct data as much as possible but there were some cases in which we had to remove the incomplete data that could not be provided where it was required. For example, mentioning non-categorized indicators when looking at the kill chain pattern taken by a user, can be distracting for the analyst. However, in the reports such as total number of indicator events per date, this type of missing data could be tolerated.

Phase 3: Explore

In this phase, we explored how the analysts achieve the detection goal and tried to add questions and features to the visualization tool so that they could find their required information more easily and more quickly. The result of this phase was a set of new questions and features added to the visualization goals list.

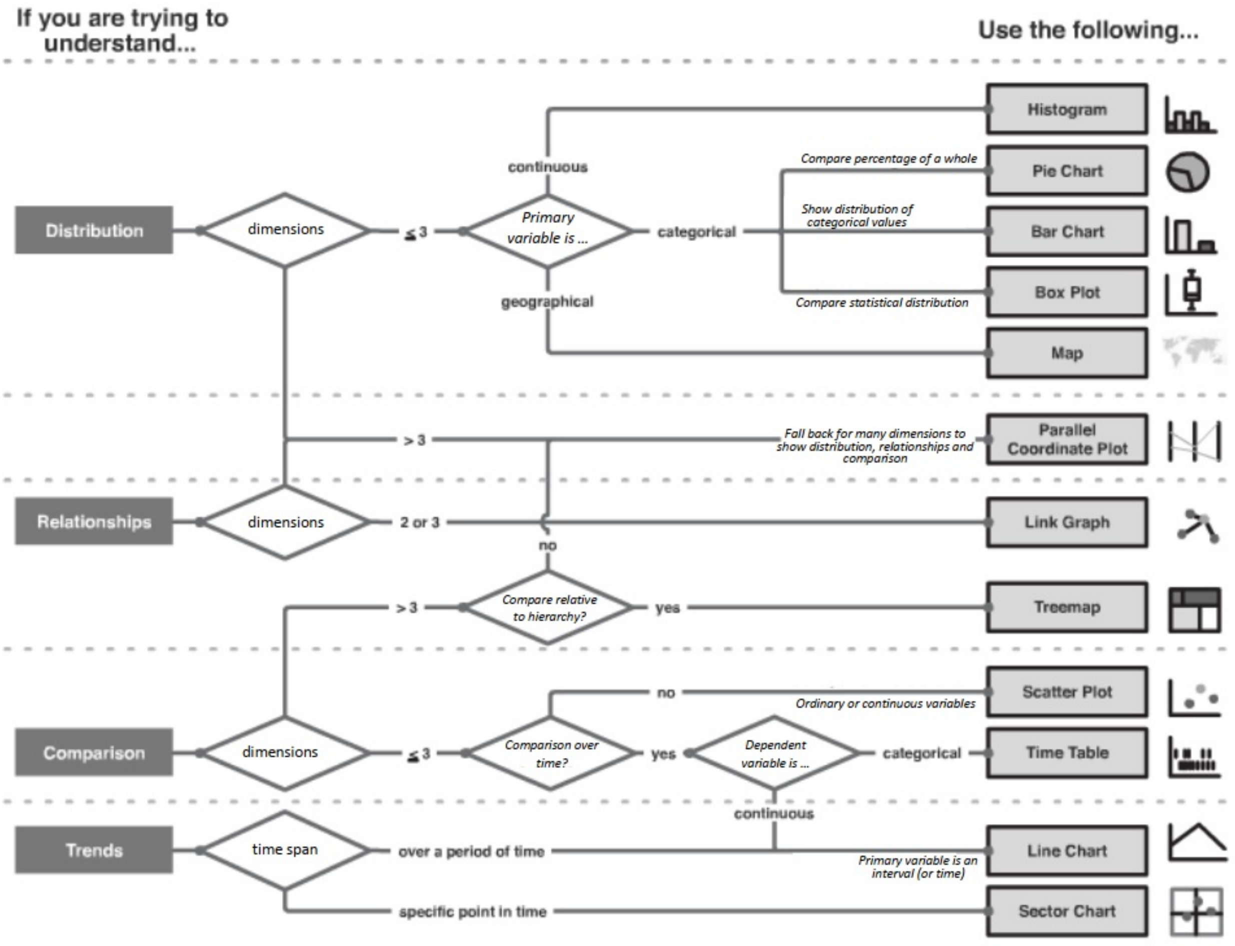
One of the new reports discovered in this phase and added to the dashboard was a schema of total number of indicators per date. This report (VQ6) shows a whole image of the day and allows a high level comparison between different dates.

Continuing providing general statistical information, we decided to briefly report total number of distinct events, indicators and kill chain steps in the selected time window (VQ7).

Looking at the observed true positive cases, we found that the execution step of the kill chain is highly considered by the analysts for insider threat detection. Therefore, although it was not mentioned in the questions list for the ontology design process, we added a report of the users with highest exfiltration score to the visualization goals (VQ8).

Phase 4: Visualize

There are different best practices and guidelines for security visualization summarized in [7]. We used these resources to find the best visualization type according to the nature of data and the way the analysts need to use it. For example, according to the flow chart shown in Figure 5.8, the best chart type to demonstrate the distribution of generated indicators and kill chain steps is pie chart since it is required to analyze the percentage of a whole.



If you are trying to understand...
Use the following...
Distribution
Relationships
Comparison
Trends
dimensions
≤3
>3
2 or 3
Primary variable is ...
continuous
categorical
geographical
Compare percentage of a whole
Show distribution of categorical values
Compare statistical distribution
Fall back for many dimensions to show distribution, relationships and comparison
Histogram
Pie Chart
Bar Chart
Box Plot
Map
Parallel Coordinate Plot
Link Graph
no
Compare relative to hierarchy?
yes
Treemap
Comparison over time?
Dependent variable is ...
Ordinary or continuous variables
Scatter Plot
Time Table
time span
over a period of time
specific point in time
Primary variable is an interval (or time)
Line Chart
Sector Chart

Figure 5.8 Visualization chart types [7, 8]

Based on Figure 5.8, we selected the visualization chart types shown in Table 5.9.

Table 5.9 Visualization chart types

Index	Chart type	Goal description
VQ1	Bar chart	Who are the top 10 users with the highest risk factor?
VQ2	Pie chart	What are the top 10 most generated indicators?
VQ3	Pie chart	Which step of the kill chain is taken the most?
VQ4	Scatter chart	What is the pattern of kill chain steps taken by a specific user in comparison with his/her teammates?
VQ5	Bar chart	Who are the users who have taken the highest step of the kill chain
VQ6	Line chart	What are the total number of events per date in a specified time window?
VQ7	Plain text	What are the distinct number of events, indicators and kill chain steps?
VQ8	Bar chart	Who are the users with highest exfiltration score?
VF1	Feature	Date filter
VF2	Feature	Username filter
VF3	Feature	Interactive diagrams

Phase 5: Feedback

During the meetings with experts, we presented the visualized reports and received feedback for improving them. For example, we found in feedback meeting that there are some teams in which almost all employees generate lots of specific indicators since the nature of the tasks in the team requires so. For example, there are teams that are responsible to send sensitive reports to partner companies for forensic purposes. Obviously, the people in this team generate lots of exfiltration indicators. Therefore, in order to make the comparison of user activity versus his/her teammates more meaningful and clear, we separated the results based on kill chain step. In other words, for each kill chain step, the analyst can monitor how the user is behaving in comparison with the team.

In addition, according to the feedback, new features were defined to make reports more usable. For instance, we had much more non-behavioral indicators every day which did not allow the changes in the number of behavioral ones be obviously detected. To solve this problem, we added a new feature (VF4) to filter indicators based on their nature. The complete list of questions and features after the last iteration is provided in Table 5.10.

5.3.2 Implementation

After defining the reports and their related chart types, we chose python as the development language since it is a very powerful language with huge data analysis capabilities while being simple, free and open source. To develop a graphical dashboard we used a python library named Dash [109]. In order to connect to GraphDB for retrieving data and visualizing it, we used the SPARQLWRAPPER [108] python library.

The dashboard includes two main tabs: general statistics and kill chain statistics. Both tabs share the same filters (Figure 5.9): start date, end date, username and indicator type (behavioral and non-behavioral). If no username is selected, the data related to all usernames will be demonstrated. Also, the analyst could select whether he/she needs to see the statistics considering only behavioral indicators, non-behavioral ones or both of them.

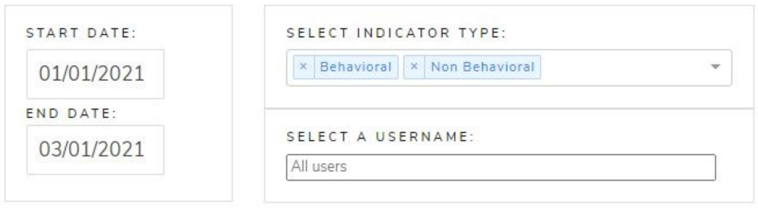

START DATE: 01/01/2021
END DATE: 03/01/2021
SELECT INDICATOR TYPE: × Behavioral × Non Behavioral
SELECT A USERNAME: All users

Figure 5.9 Filters shared by dashboard tabs

Based on the values of the aforementioned filters, a summary of the information is shown at the top of the dashboard including the total distinct number of events, indicators and kill chain steps taken (Figure 5.10).

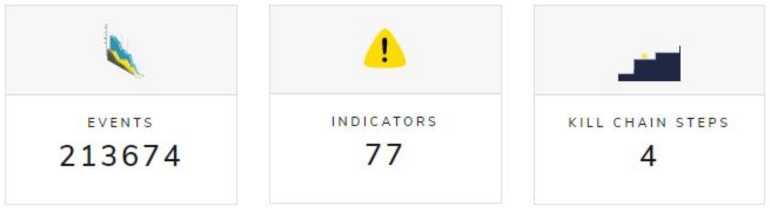

EVENTS 213674
INDICATORS 77
KILL CHAIN STEPS 4

Figure 5.10 A summary of the information shown in tab-1

In the general statistics tab, we could find the pattern of total number of indicator events

Table 5.10 Visualization goals in the last iteration

Index	Goal type	Goal description
VQ1	Question	Who are the top 10 users with the highest risk factor?
VQ2	Question	What are the top 10 most generated indicators?
VQ3	Question	Which steps of the kill chain are taken the most?
VQ4.1	Question	What is the pattern of kill chain steps taken by a specific user
VQ4.2.1	Question	What is the pattern of generating indicators related to the first step of the kill chain by a specific user in comparison with his/her teammates?
VQ4.2.2	Question	What is the pattern of generating indicators related to the second step of the kill chain by a specific user in comparison with his/her teammates?
VQ4.2.3	Question	What is the pattern of generating indicators related to the third step of the kill chain by a specific user in comparison with his/her teammates?
VQ4.2.4	Question	What is the pattern of generating indicators related to the fourth step of the kill chain by a specific user in comparison with his/her teammates?
VQ4.2.5	Question	What is the pattern of generating indicators related to the fifth step of the kill chain by a specific user in comparison with his/her teammates?
VQ5	Question	Who are the users who have taken the highest step of the kill chain
VQ6	Question	What are the total number of events per date in a specified time window?
VQ7	Question	What are the distinct number of events, indicators and kill chain steps?
VQ8	Question	Who are the users with highest exfiltration score?
VF1	Feature	Date filter
VF2	Feature	Username filter
VF3	Feature	Interactive diagrams
VF4	Feature	Indicator type (behavioral/non-behavioral) filter

generated per date (Figure 5.11) and identify the most generated indicators and kill chain steps by a specific user or all of them (Figure 5.12 and 5.13).

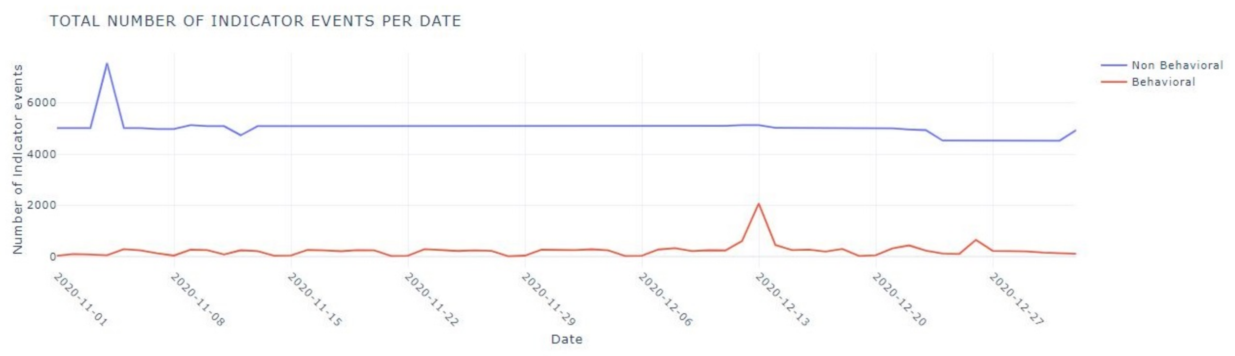


Figure 5.11 Total number of behavioral and non-behavioral indicators per date by all users from 2020-11-01 to 2021-01-01

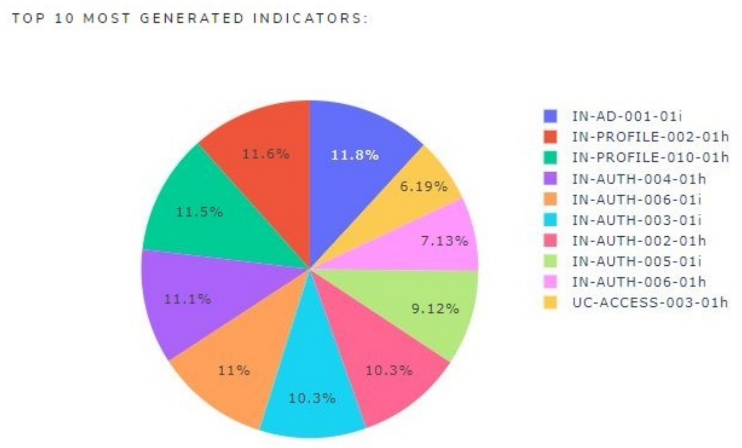


Figure 5.12 Top 10 most generated indicators by all users from 2020-11-01 to 2021-01-01 considering both behavioral and non-behavioral indicators

All the predefined filters could be used to change these diagrams. For example one might be interested in only seeing the pattern of behavioral indicators generated by a specific user. By hovering mouse cursor on any point, more detailed information will appear. Furthermore, analyst could zoom in the first diagram (Figure 5.9) to find the exact date related to a peak point. In addition, in cases that an indicator or a kill chain step is generated much more than others, analyst can remove the frequently happening item by clicking on its name from the legend list of the diagrams shown in Figure 5.12 and 5.13 to be able to look at the statistics related to other indicators or kill chain step.

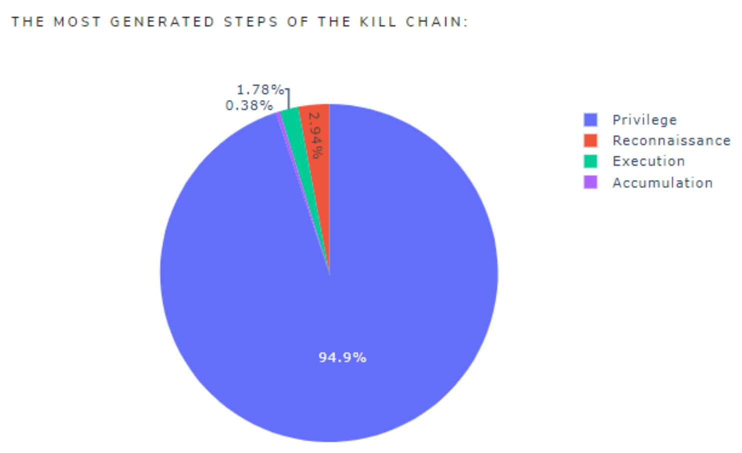


Figure 5.13 Top most generated steps of the kill chain by all users from 2020-11-01 to 2021-01-01 considering both behavioral and non-behavioral indicators

In the second tab, we are mostly focused on detecting the suspicious insiders and their kill chain activity. High risk users are sorted based on three different points of view: highest risk factor, highest step of the kill chain reached and highest exfiltration score. In the first diagram demonstrated in Figure 5.14, we could see the users who get the highest risk factor in the specified time window (which can be calculated considering only behavioral indicators, non-behavioral ones or both of them).

Second diagram shown in Figure 5.15, lists top 10 users who have reached to the highest step of the kill chain (here the highest step of the kill chain a user could reach is step 4). For seeing each user's kill chain activity, the analyst could simply click on his related bar in the diagram or select his/her username from the filter box.

According to the procedures the analysts take while investigating user's behavior, we found out exfiltration data is the most important step of the kill chain. Therefore, the last diagram in this row is designed to identify top 10 users who get the highest exfiltration score (Figure 5.16). This list helps to detect users who were not listed in the other reports as they did not follow a kill chain pattern nor generate high amount of indicators.

We can select a user from these three diagrams or from the username box to see the kill chain activity of the user as shown in Figure 5.17.

Finally, in order to decrease the rate of false positives, we can see the user behavior in comparison with his/her teammate activities. As demonstrated in Figure 5.18, we separate user behavior based on the kill chain step and compare the number of IoCs generated by the user versus the average number of IoCs generated by other teammates. Again, the username

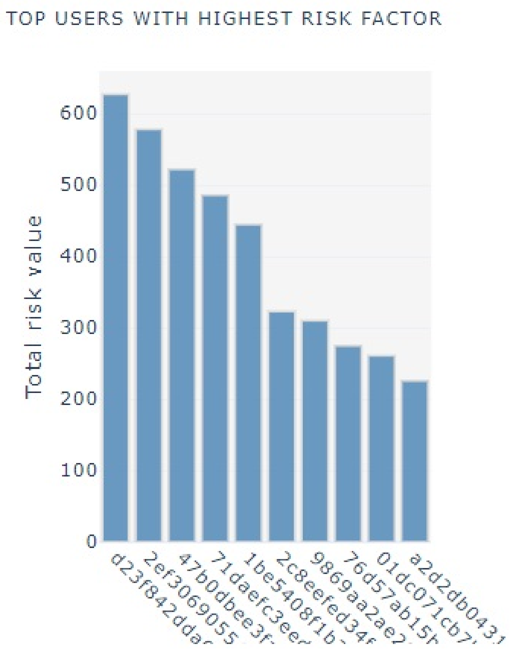


Figure 5.14 Top 10 users with highest risk factor from 2021-01-01 to 2021-03-01 considering both behavioral and non-behavioral IoCs

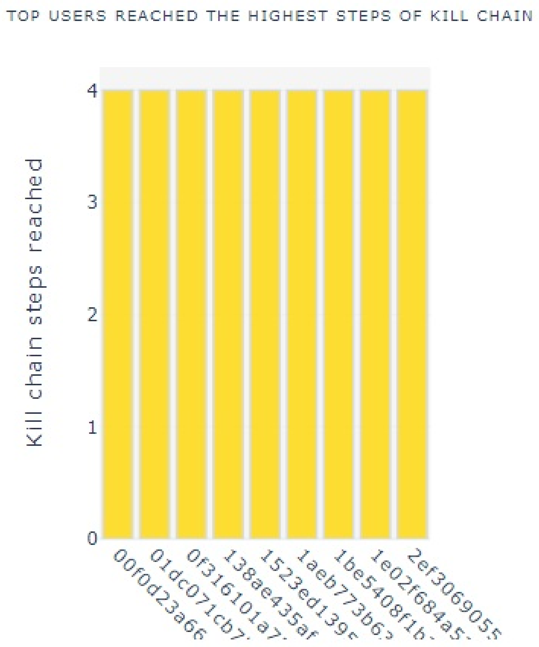


Figure 5.15 Top 10 users who have reached the highest step of the kill chain

can be selected from the reports shown in Figure 5.14, 5.15 and 5.16 or can be chosen from

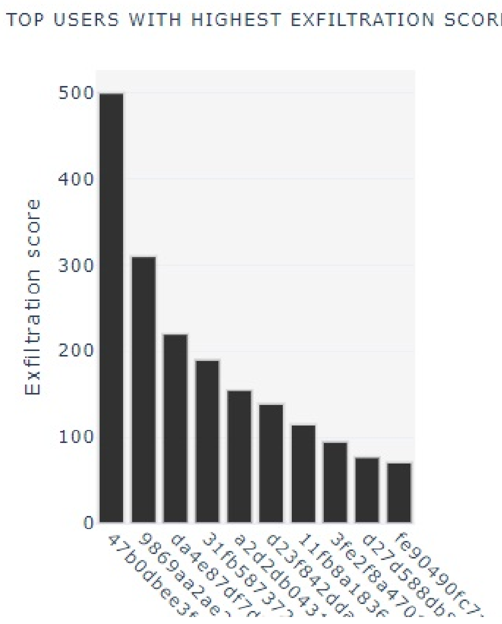


Figure 5.16 Top 10 users with highest exfiltration score from 2021-01-01 to 2021-03-01 considering both behavioral and non-behavioral IoCs

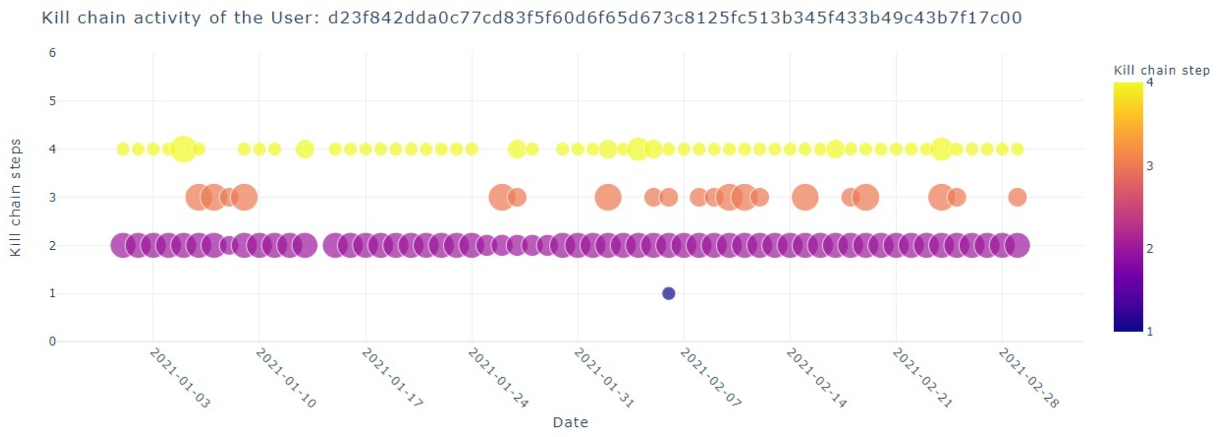


Figure 5.17 The kill chain activity of a sample user from 2021-01-01 to 2021-03-01 considering both behavioral and non-behavioral indicators

the username box in the filter bar.

In the example shown in Figure 5.18, we can see that the user behavior in generating IoCs related to the first and second steps of the kill chain is not very different from his/her teammates. It might indicate the tasks in this team include activities that may trigger indicators of steps 1 and 2. Looking at the third step, the user might seem suspicious as

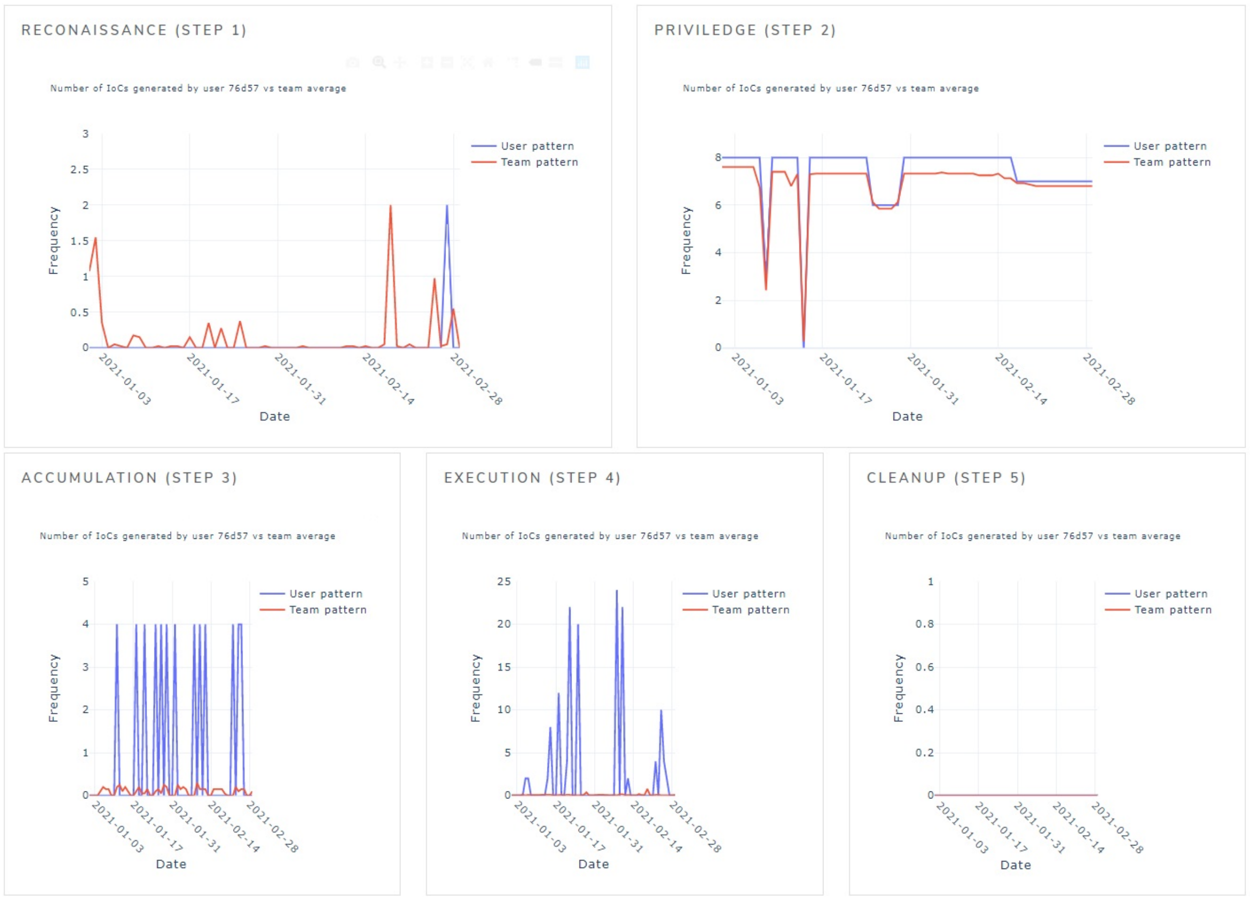


Figure 5.18 The kill chain activity of a sample user from 2021-01-01 to 2021-03-01 versus his/her teammates

he/she is generating higher amount of IoCs versus his/her teammates. However, considering the number of IoCs, it may not need to be investigated. Finally, the user activities related to the fourth step of the kill chain does not seem normal as the team average is much lower.

5.4 Summary

In this chapter, we first demonstrated the high level architecture of IVIIO which contains two main parts: ontological solution and visualization tool (Figure 5.1). Then, we described the methodology and implementation details of IVIIO in the next sections.

First, we explained the ontology design and implementation process. The ontology was designed following ATOM methodology that suggests six main steps to reach the final version of the ontological model (Figure 5.2): 0) Defining queries in natural language 1) Translating the natural language queries to SPARQL Queries 2) Identifying raw data specification 3)

Defining intermediate steps 4) Evoking translation rules and 5) Enriching the ontology. Since in this research we used ATOM in a real environment, there were some challenges to be addressed listed in the following.

1. **Raw data specification is needed for writing SPARQL queries.**
   We found it hard to write SPARQL Queries without knowing about the available data. Therefore, we rewrote the SPARQL Queries after identifying raw data specification.

2. **There might not be any intermediate step.**
   We noticed that having intermediate steps is not always the case. After discussions with Desjardins analysts and defining the desired questions to be answered by our solution, we could not end up with a very complicated query that has multiple intermediate steps while they are still important for the investigation process.

3. **It should be specified how to use predefined ontologies.**
   Currently, the process of reusing other ontologies is not discussed in the guideline of ATOM while this feature is very useful. We decided to use the main concepts defined in the ontologies designed by previous students such as the works done by Sadighian [20] and Ducharme [22] as super classes for the classes in our ontology.

4. **More detailed guideline is needed.**
   There is not enough detailed information (e.g. how to choose related sensors, how to find raw data sources related to each query, how to divide query to sub-queries, etc.) to follow the steps.

After explaining the design process, we briefly described how the ontological model developed by Protégé was imported in GraphDB and got populated. Before population, we normalize input logs to minimize number of queries during the population process.

In the last section of this chapter, we went through the design and implementation details of the visualization tool. There are many advantages of visualization such as:

- Visualized reports can transfer high amount of information in short time.
- Trends and patterns are easily detectable.
- There is no need any specific knowledge to query the database
- It helps to decrease false positive events since a human can decide about the incidents by looking at the patterns and find the similarities or connection.

- Potential attacks can be detected in early stages.
- It is customizable using filters.

With the aim of helping analysts to reduce their investigation time, we developed an interactive dashboard including different filters affecting the diagrams so that the analysts can see the required data without knowing how to query the ontological database. We deployed the procedure proposed by Balakrishnan in [7] (Figure 5.7) for developing this visualization tool. Every step is described and the important outcomes such as visualization goals are mentioned in detail (Table 5.10).

We used python as the programming language and SPARQLWRAPPER and Dash libraries to implement the visualization tool. The main characteristics of the dashboard are also described. To summarize, it can be mentioned that the dashboard includes two main tabs and contains nine different reports:

1. Summary of distinct number of events, IoCs and kill chain steps
2. Total number of IoC events per date
3. Top 10 most generated IoCs
4. The most generated steps of the kill chain
5. Top 10 users with highest risk factor
6. Top users who have reached the highest step of the kill chain
7. Top 10 users with highest exfiltration score
8. Detailed kill chain activity of the user
9. Comparison of user activity versus his/her team average considering each step of the kill chain

These reports can be updated based on different filters provided in each tab including time window, indicator types (behavioral, non-behavioral or both of them) and username. For the convenience of the analyst and avoid waste of time, the last two reports (number 8 and 9) can also be updated by selecting a user listed in the reports number 5, 6 and 7 (by clicking on the related bar). Other features such as zooming in and out, changing the scale, downloading the plot, etc. are also provided.

CHAPTER 6 EVALUATION

We deployed two approaches to evaluate our solution. First, three synthetic scenarios were developed by another labmate and a research assistant in SecSi lab. Second, to make the evaluation process stronger, we asked Desjardins to provide two observed scenarios and analyzed IVIIO against them. In the remaining of this chapter we describe the scenario generation process and take a look at each scenario by demonstrating how IVIIO can ease the insider indicator investigation process.

6.1 Assumptions

In the synthetic scenarios, we assume that all information is collected and there is no missing data due to a policy, device or network malfunction. For example, if an activity related to a step of the kill chain is not reported, there is no way for IVIIO to detect a complete kill chain activity. The process of solving these kinds of problems is out of the scope of this project.

The other assumption is related to the nature of kill chain attacks and asserts the order of kill chain steps taken in a kill chain attack is not necessarily strictly ascending. This assumption is made regarding the observed cases analyzed by the experts in Desjardins. To make this assumption more clear, consider a user behavior to be shown by the kill chain steps he/she is taking.

- user_1: 1,1,2,1,3,2,4,1,5
- user_2: 1,1,2,1,4,2,4,1,5
- user_3: 3,2,2,1,3,2,4,1,5

We consider the behavior of user_1 as a kill chain activity since an ordered sub-sequence of 1,2,3,4,5 could be found in the main sequence. We do not call user_2 as a kill chain attacker since there is no activity related to the 3rd step. Additionally, regarding user_3 behavior, as there is no ordered sub-sequence containing first to fifth step respectively, we do not report a kill chain attack.

Thus, although a user could jump back to the previous kill chain steps in the middle of an attack, an ordered sequence of kill chain steps from the first step to the last one should be taken respectively in a complete kill chain attack. The example also reveals the fact that no kill chain attacker can skip a step. As seen in user_2 behavior, the user is generating IoCs related to all kill chain steps except the third one. Therefore, he/she cannot be introduced as doing a kill chain attack even if the order of events follows the kill chain pattern.

The third assumption is that we do not expect IVIIO to report all suspicious users in a long time window as only the top 10 high risk users are reported. We assume IVIIO is used daily and the behaviors of the high risk users listed in each day are supposed to be investigated with the help of other complementary reports such as user kill chain activity, team pattern, top indicators generated, etc.

Finally, we assumed all the users to work in a team. In the cases the user's team is not mentioned, we considered the user is working in a one-person team including only him/herself.

6.2 Scenario generation process

As mentioned before, we used three synthetic scenarios developed by our teammates. In this section, we describe the process they followed for generating these scenarios.

6.2.1 Phase 1: Indicator categorization

Since each scenario needs to contain a huge amount of user activities, writing them manually would have been very time consuming. Therefore, the scenario design team decided to automate part of the process by coding a script. According to the fact that randomly selecting indicators results in a non-realistic scenario, the script selects indicators considering the kill chain step they belong to and their sensitivity level. Also, it is required to know whether an indicator is showing an activity (such as modifying a file) or is just a privilege status (such as being a high privilege user). Currently, both types are known as indicators and there is no categorization separating privilege statuses and user activities.

For defining sensitivity, the indicator assumed to be normal or abnormal based on the fact that whether it shows something happening usually in a normal behavior (such as belonging to an active directory group) or it demonstrates a non-usual behavior (such as sending sensitive files to a personal email). To provide this information, by looking at the description of indicators, the scenario design team categorized all the indicators in 11 different groups, considering the aforementioned characteristics. Table 6.1 shows the information of these 11 groups.

For instance, all of the indicators related to the first step of the kill chain that could be normal by themselves and show privilege statuses, belong to the first group named rPN.

Table 6.1 Indicator categories defined by the scenario design team

Index	Groupe name	Kill chain step	Privilege status	Status
1	rPN (reconPrivNormal)	Reconnaissance	Yes	Normal
2	aAN (autreActivNormal)	Other	No	Normal
3	pPN (privPrivNormal)	Privilege	Yes	Normal
4	pAN (privAccumNormal)	Privilege	No	Normal
5	rAA (reconActivAnormal)	Reconnaissance	No	Abnormal
6	pPA (privPrivAnormal)	Privilege	Yes	Abnormal
7	pAA (privActivAnormal)	Privilege	No	Abnormal
8	aAA (accumActivAnormal)	Accumulation	No	Abnormal
9	eAA (exfilActivAnormal)	Exfiltration	No	Abnormal
10	cAA (cleanActivAnormal)	Clean	No	Abnormal
11	tAA(autreActivAnormal)	Other	No	Abnormal

This categorization helps the code to define step by step kill chain activity or other malicious behaviors without knowing the exact description of the indicator. It also helps to simulate a false positive situation in which a normal user generates many non-critical indicators.

6.2.2 Phase 2: Indicator selection

Now that all the indicators are categorized into 11 groups, one could define a user behavior in a time window by the number of generated indicators in each group. The user behavior is shown by a set of sequences:

$$User\_behavior = S_1 S_2 ... S_n$$

in which, each sequence contain a set of indicators and S_i happens before S_j if i<j. In other words, the indicator events in each sequence have a timestamp less than or equal to the next sequence. Each sequence is defined by a duodecuple (12-tuple) shown as:

$$S_i = (I_i, |rPN|, |pPN|, |aAN|, |pAN|, |pPA|, |eAA|, |aAA|, |rAA|, |tAA|, |pAA|, |cAA|)$$

where I is the number of iteration and |x| is the number of indicators in the group “x”. The script will look at the duodecuples that specify user behavior and select sample indicators in each group randomly. Due to the fact that the nature of the indicators in the same group is almost the same, a randomly selected indicator has the characteristics shared among all the other indicators in the same group. Therefore, this usage of randomness will not disrupt the scenario story. Thus, we can simulate a realistic scenario by defining the general behavior manually and instantiating indicators automatically.

For the purpose of not being biased, the scenario generation team used different iterations to select indicators. In each iteration, the predefined number of indicators are selected and added to the scenario. For example, in the sequence S1=(5,0,5,0,3,0,0,0,6,0,0,0), there are five iterations. In each iteration, 14 indicators (5 from pPN group, 3 from pAN group and 6 from rAA group) are listed. Hence, in the final result, the user behavior consists of 5 sets of 14 indicators happening in different dates. Using iterations helps to have more realistic scenario as users usually tend to follow a repetitive pattern. In the cases where we need a burst non-usual behavior, we can lower the number of iterations so that the required indicators happen in one or two days only. There is also an option to have an ordered list of indicators when a kill chain activity is simulated.

The scenario design process can be summarized in the following steps:

1. Define general user behavior using different sequences (S1,S2,...).
2. Define number of iteration for each sequence.
3. Define number of indicators in each group.
4. Set the start and end date stamps.
5. Set the indicators inside a sequence to be ordered or disordered.

By taking the aforementioned steps, we can design the big picture by selecting how the user is behaving in terms of taking kill chain steps and generating normal or abnormal indicators. In the next step, to have a scenario in the desired format, we use the script to find indicator samples for the defined behavior.

Using the described process, three different scenarios are generated to evaluate IVIIO which are described in the following sections.

6.3 Evaluation scenarios

In this section we will describe each scenario and evaluate IVIIO fed by three synthetic and two observed scenarios to see if it could help the analyst to detect and investigate the threats.

6.3.1 Scenario no.1

Since Desjardins insider threat detection team does not currently focus on kill chain attacks, having such real labeled data was not feasible. Therefore, we decided to generate a scenario in which a user is doing a complete kill chain attack.

Description

The first scenario shows the activity of a malicious user named "user_x" who takes the complete kill chain steps in order. A summary of the scenario is mentioned in the following:

- Number of indicator events: 205
- Number of distinct indicators: 58
- Number of distinct dates: 16
- Number of distinct users: 1

The user behavior is generated by concatenating three sequences:

- S1 = (10,2,5,1,1,0,0,0,0,0,0,3)
- S2 = (1,0,0,0,0,5,10,3,3,2,1,1)
- S3 = (5,1,10,1,1,0,0,0,0,0,0,0)

It is clearly noticeable that first and third sequences include normal activities while the second one simulates a kill chain attack.

Investigation

By adding indicator events in this scenario to the ontological database, the queries for generating high risk reports (VQ1, VQ4 and VQ8) execute. Therefore, one could see "user_x" appearing in two reports: top users with highest risk factor (Figure 6.1) and users who have reached to the highest step of the kill chain (Figure 6.2).

As this user is flagged as the top suspicious users in two reports, his behavior should be investigated. By clicking on the bar related to "user_x", we can see his kill chain activity shown inside the red area in Figure 6.3 which is generated by VQ4.1 query.

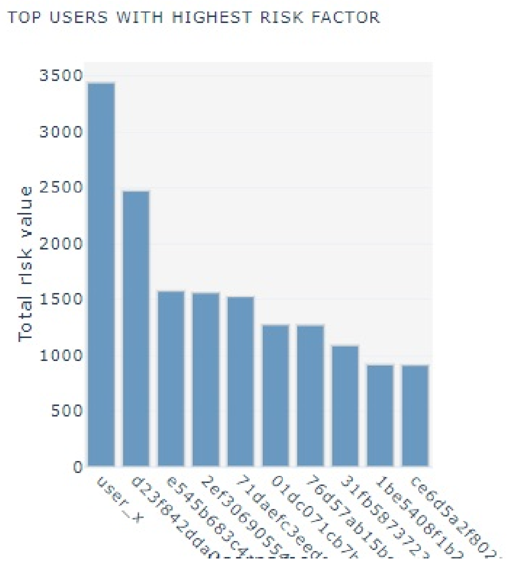


Figure 6.1 Top users with highest risk factor from 2020-01-01 to 2021-07-31

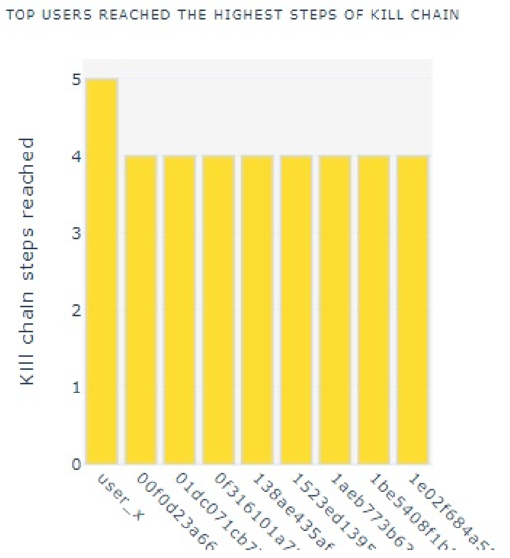


Figure 6.2 Top users reached to the highest step of the kill chain

Now that we discovered this user has taken all the five steps of the kill chain, let us take a look at the most generated indicators in the date he has taken all five steps. Because of the connection between each event and its details on our ontological model, by hovering mouse on the dots in Figure 6.4, the details of the events will appear and we see that the date all this has happened was 2021-07-15. Now we are eager to investigate user activity on 2021-07-15. Therefore, we type “user_x” in user filter box and set the start and end date

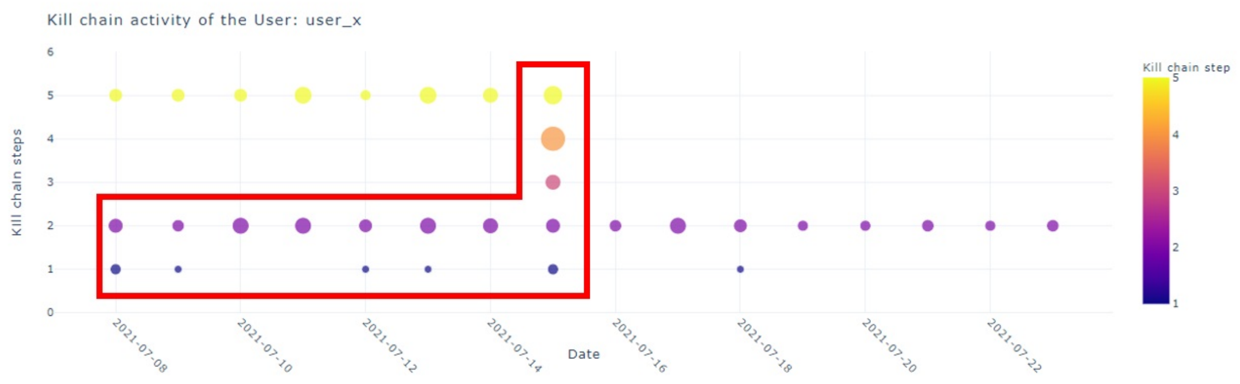


Figure 6.3 Kill chain activity of “user_x”

equal to 2021-07-15. The result is shown in Figure 6.4 generated by queries VQ2 and VQ3 that query the related instances of the :IndicatorEvent class.

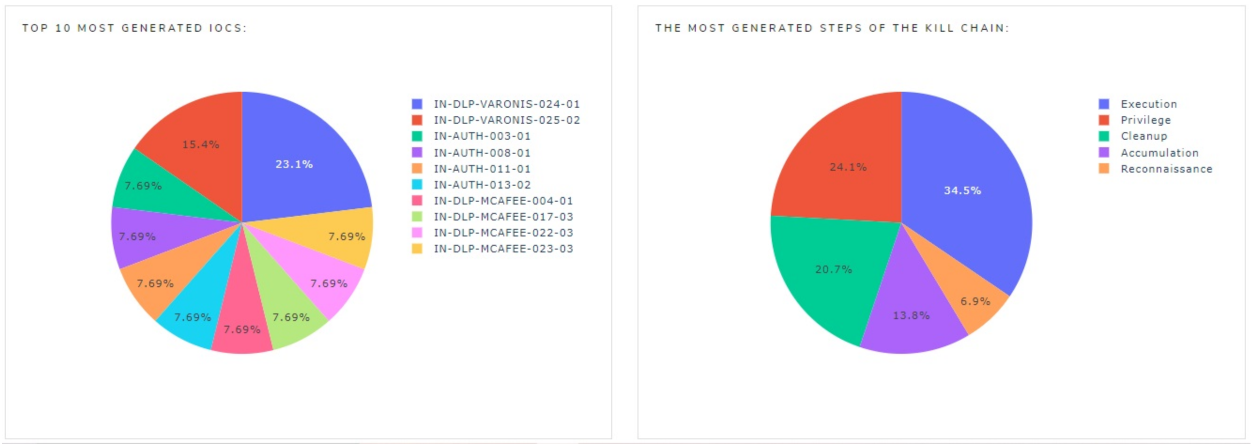


Figure 6.4 Most generated IoCs and kill chain steps by “user_x” on 2021-07-15

Figure 6.4 shows there are unimportant indicators that could be ignored by selecting behavioral indicators in the indicator type filter. Therefore, the reports will look like Figure 6.5 again generated by queries VQ2 and VQ3. For filtering indicator events to see only behavioral ones, we can easily query :BehavioralIndicatorEvent class which is a subclass of the class :IndicatorEvent.

Looking at Figure 6.5, there are many behavioral indicators generated by “user_x” on 2021-07-15. We will take a look at the descriptions of these indicators in Table 6.2. It should be noticed that the analysts have all this information and could look at the name of indicator and understand what the user was going to do.

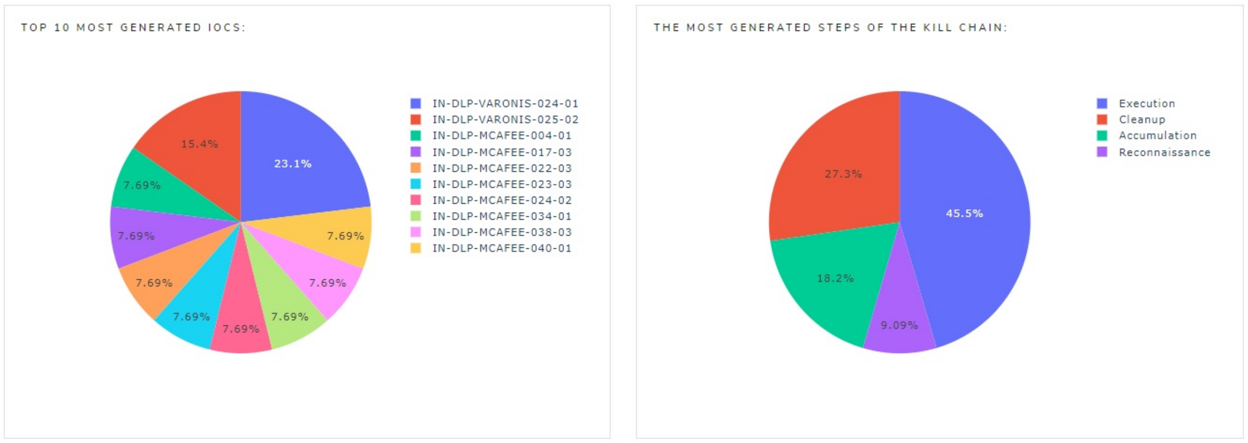


Figure 6.5 Most generated IoCs and kill chain steps by “user_x” on 2021-07-15 considering only behavioral indicators

Result

As explained in Table 6.2, the user has tried to exfiltrate credit card and SIN information using different channels such as USB, uploading to a cloud or web and printing. Generating 10 exfiltration event in a single day shows this user highly dangerous and the appropriate actions should be taken.

6.3.2 Scenario no.2

For testing IVIIO against normal users who may seem suspicious, our teammates designed a scenario in which a normal user generates many random unimportant indicators.

Description

In this scenario we received 175 indicator events generated by a single user named “user_y”. The details of input file is summarized below:

- Number of IoC events: 175
- Number of distinct indicators: 45
- Number of distinct dates: 16
- Number of distinct users: 1

Table 6.2 Description of IoCs generated by the suspicious user in scenario no.1

Index	IoC description
1	Abnormal behavior: unusual number of files deleted
2	Abnormal behavior: unusual number of sensitive files deleted
3	Upload to a cloud storage service (Dropbox, GoogleDrive, iCloud, O365, etc.)
4	Block-Printer
5	Copy to USB - small number of credit cards
6	Copy to USB - small number of credit cards
7	Copy to USB - large number of credit cards
8	Upload to the Web - small number of credit cards
9	Upload to the Web - large number of credit cards
10	Upload to the Web - large number of SIN numbers

The sequences used to simulate the behavior of user_y are listed here:

- S1 = (6,2,5,3,1,0,0,0,0,0,0,0)
- S2 = (1,0,0,0,0,1,1,0,0,0,0,1)
- S3 = (5,1,10,1,1,0,0,0,0,0,0,0)
- S4 = (1,0,0,0,0,1,0,1,0,0,0,0)
- S5 = (3,2,8,2,3,0,0,0,0,0,0,0)
- S6 = (1,0,0,0,0,1,1,0,0,0,0,1)

Looking at the sequences one could see there are only a few suspicious activity among many privilege statuses and unimportant indicators.

Investigation

After populating the ontological database with the information of new events generated by “user_y”, we could see his name among suspicious users listed in two reports: high risk users and users reached to the highest step of the kill chain (Figure 6.6) which are generated using VQ1 and VQ4 queries.

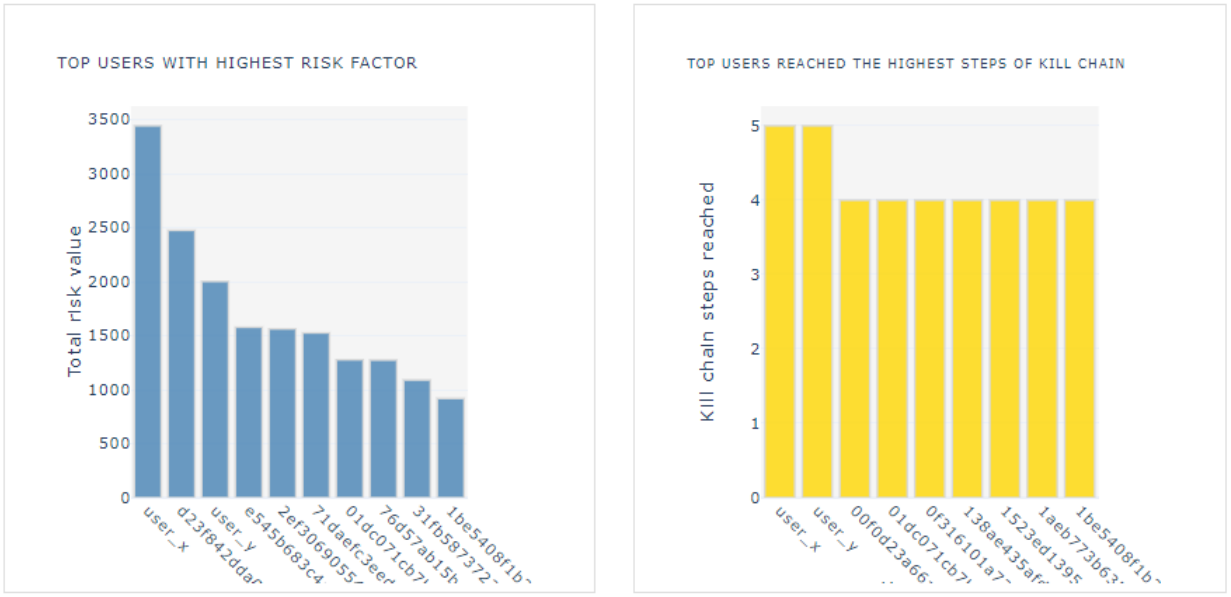


Figure 6.6 Top users with highest risk factor and highest step of the kill chain reached after appending information of scenario no.2

By clicking on the related bar in the charts shown in Figure 6.6, we could see the user kill chain activity as the result of query VQ4.1, shown in Figure 6.7.

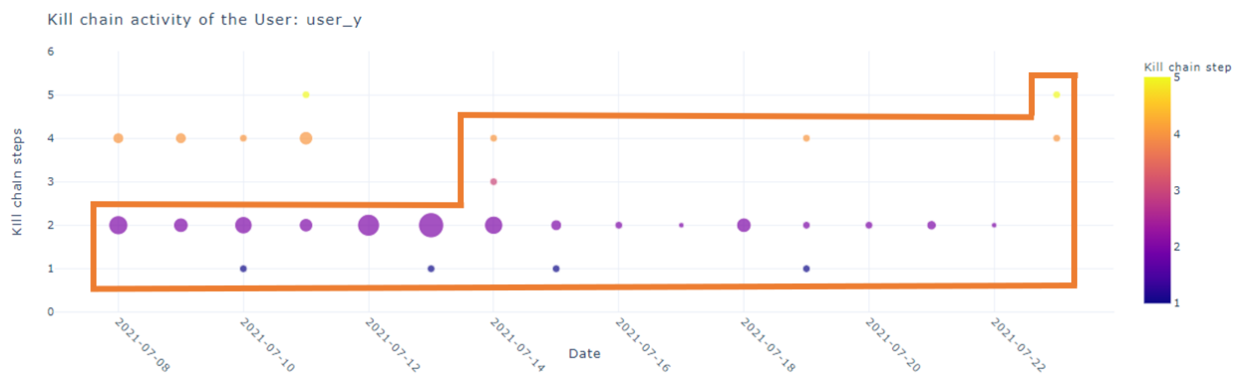


Figure 6.7 Kill chain activity of “user_y”

Let us inspect the IoCs generated by the user. For this purpose, we type username in the related filter and select the date window to the dates exfiltration happened (2021-07-08, 2021-07-09, 2021-07-10, 2021-07-11, 2021-07-14, 2021-07-19, 2021-07-23). Query VQ2 receives these values as its filters and we will look at the behavioral and non-behavioral indicators to compare user privileges versus his actions. In order to be able to discuss all the data in the investigated dates, we summarized the most relevant IoCs in Table 6.3.

Table 6.3 Investigating the IoCs generated by the suspicious user in scenario no.2

Date	Privilege IoC description	Behavioral IoC description	Result
2021-07-08	User has access its USB ports	Copy to USB	FP
20201-07-09	user has access its USB ports	Copy to USB	FP
20201-07-10	User has access its USB ports	Copy to USB	FP
20201-07-11	User doesn't have access the USB ports	Copy to USB and upload on web	Should be investigated
20201-07-14	Other privileges related to printing and business rights	Access the USB port while not being authorized to, write on disk	Should be investigated
20201-07-21	User has access its USB ports	Copy to USB	FP

Result

As mentioned in Table 6.3, user_y has copied some files to USB drives on 2021-07-14 and 2021-07-21 while was not authorized to do so on these dates. However, these privilege IoCs do not express a complete profile of the user. Other exfiltration events may be ignored based on the analyst idea as the user had the required authorization in some days. Therefore, the user seems not to be malicious while still needs a more detailed investigation.

6.3.3 Scenario no.3

In the third scenario some specific indicators are selected manually to show a potential malicious behavior. The rest of scenario is generated in the same way the previous ones were done.

Description

This scenario contains 216 indicator events happening in 7 different dates by a user named "user_z". The general information of the scenario is provided below:

- Number of IoC events: 216
- Number of distinct indicators: 44

- Number of distinct dates: 7
- Number of distinct users: 1

The specific indicators selected by the scenario design team were about acquiring a high privilege account, accumulating credit card information and sending sensitive files to an outside email address. If we show the list of these specific indicators by L = {ind_1, ind_2, ind_3, ind_4, ind_5}, the whole scenario is generated using the sequences listed below:

- S1 = L[0] , (10,2,5,3,1,0,0,0,0,0,0,0) , L[1]
- S2 = (5,1,10,1,1,0,0,0,0,0,0,0) , L[2]
- S3 = (3,2,8,2,3,0,0,0,0,0,0,0) , L[3], L[4]

Investigation

To evaluate how IVIIO reports "user_z", we can take a look at the high risk users reports after updating database using the new information (Figure 6.8). These reports are the results of queries VQ1, VQ4 and VQ8.

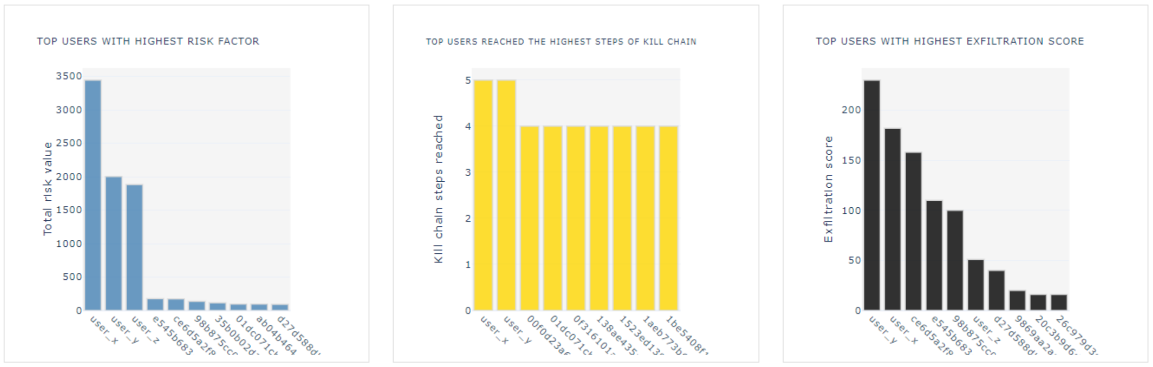


Figure 6.8 Top users with highest risk factor and highest step of the kill chain reached after appending information of scenario no.3

As shown in Figure 6.8, "user_z" is reported among the high risk users as well as users with highest exfiltration score. For looking into the details of his behavior, we click on the related bar to execute query VQ4.1 and analyze the report shown in Figure 6.9. Since in this document we are unable to discover the report by mouse action, for more clarity, we circled the information that might be hard to detect in the snapshot.

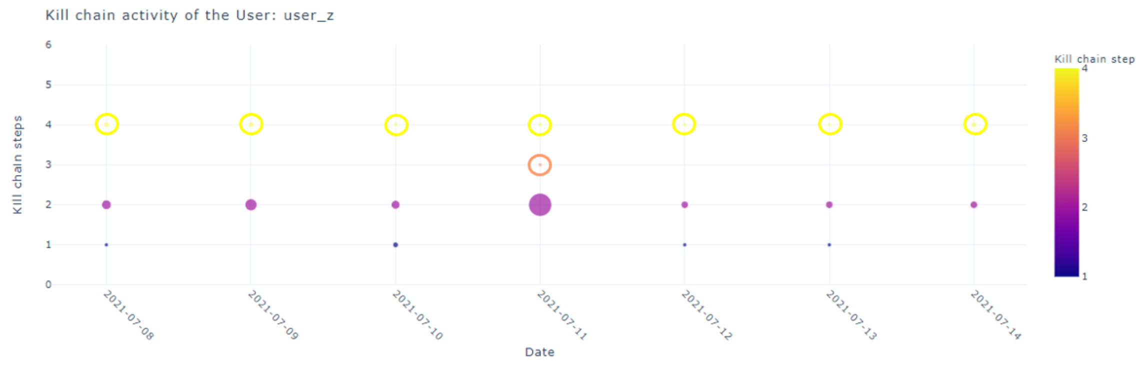


Figure 6.9 Kill chain activity of "user_z"

As seen in the diagram there are few exfiltration indicators happening every day. We can get more insight by looking at the indicators he has generated. First, considering the report generated by query VQ6, we could detect many non-behavioral indicators generated by "user_z" on 2021-07-11 (Figure 6.10). Although non-behavioral IoCs are happening frequently, generating 84 instances of them in a single day needs to be investigated.

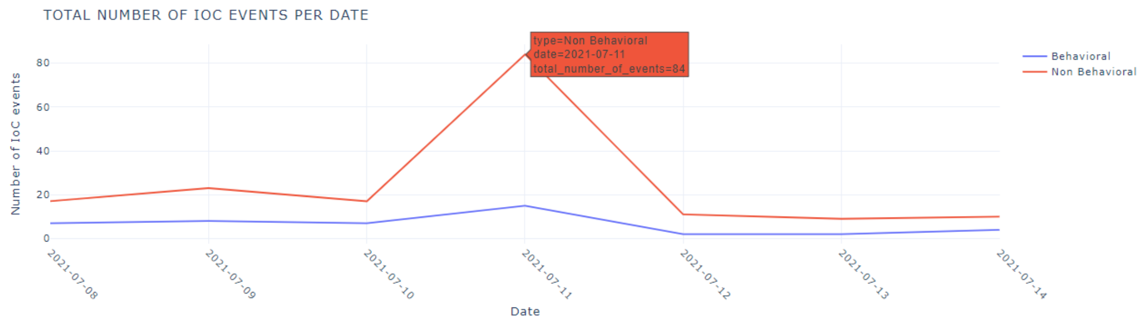


Figure 6.10 Total number of indicators generated by "user_z"

We will take a look at top 10 most generated non-behavioral indicators in Figure 6.11. This report is generated based on the result of query VQ2 considering only :NonBehavioralIndicatorEvent class. It is worth mentioning one of the most generated indicators asserts the user is a database administrator. Looking at the different non-behavioral indicators generated each day, there are some contradictory indicators (such as having and not having the right to access USB drive, having and not having high privilege accounts in the same day, having and not having access to VPN, etc.) are detected. For example, the two marked indicators listed in Figure 6.12 demonstrate this situation on 2021-07-09. One indicates the user does not have an account with high business permission rights while the other one means exactly

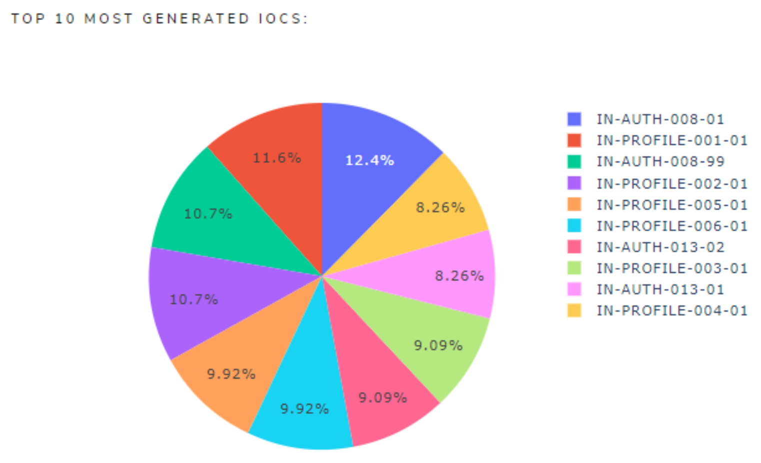


Figure 6.11 Top 10 most generated indicators generated by “user_z”

the opposite.

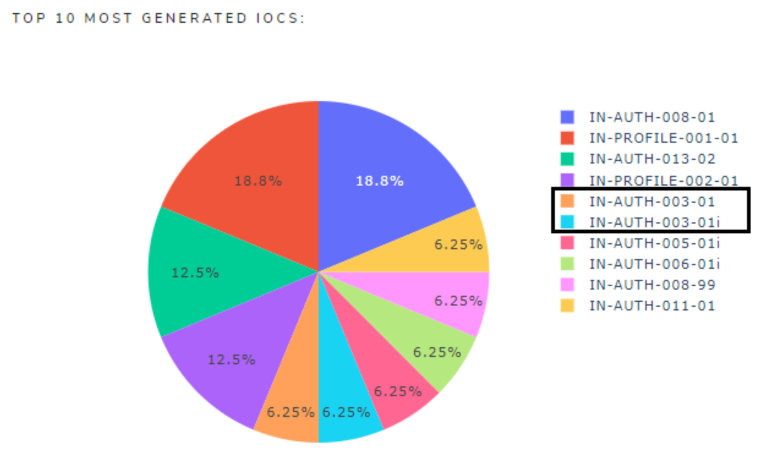


Figure 6.12 Top 10 most generated indicators generated by “user_z” on 2021-07-09

Result

Considering the indicator introduced the suspected user as a database administrator and the one related to copying files in USB drive every day, a scenario in which the user changes his own privileges to exfiltrate data might be possible. By changing privileges, he generates many non-behavioral indicators and gets reported among users with highest risk factor. Therefore, although he seems to have the privilege for using USB port, his behavior seems malicious and needs to be investigated.

6.3.4 Scenario no.4

This scenario is about an observed user in Desjardins who is detected as an potential insider threat.

Description

We were provided with 3602 IoC events related to a user (with anonymized username "`f8ceb e036cf632c47ea8ee5dc467c0b813403fe13d21beaf353199a94b19f73a`") who was detected as a potential insider threat because of modifying sensitive information and exfiltrating a file containing sensitive information. The user behavior in the scenario file is summarized in the following:

- Number of IoC events: 3626
- Number of distinct indicators: 28
- Number of distinct dates: 566
- Number of distinct users: 1

The events were collected from 2020-01-01 to 2021-07-06. We did not have the information about this user before. Therefore, at first, we populated the ontology with events information. Figure 6.13 demonstrates all the activities of this user in a glance. Looking at this diagram we can see unusual activities in behavioral indicators in May 26th.

The most generated indicators and kill chain steps by this user between 20201-05-20 to 2021-05-31 is shown in Figure 6.14.

Investigation

Now let us assume we do not know anything about this user and see if IVIIO is capable to detect him. We imagine today is 2021-05-26 and we look at the dashboard to see top 10 users who get the highest exfiltration scores from yesterday to today (Figure 6.15). This report is generated using the query VQ8. We click on every bar in this chart to take a look at the user activity and go through the details if required. Each click triggers all the queries starting with "VQ5".

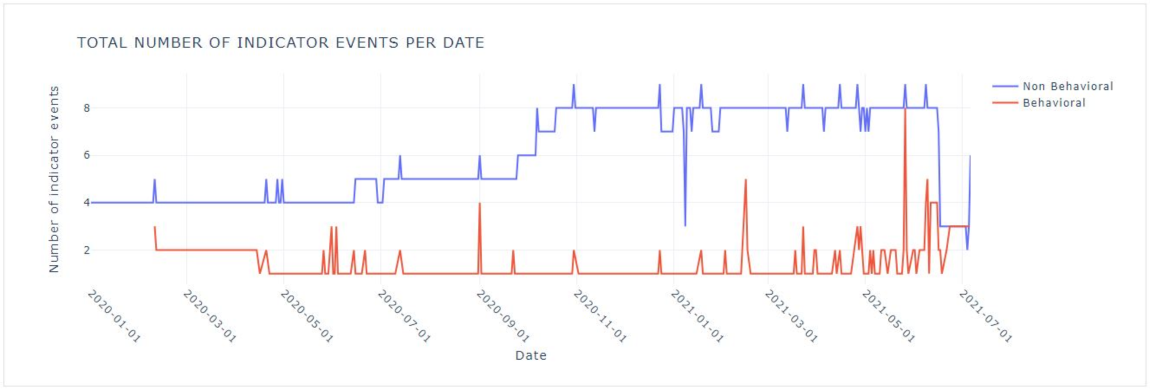


Figure 6.13 User activity from 2020-01-01 to 2021-07-06

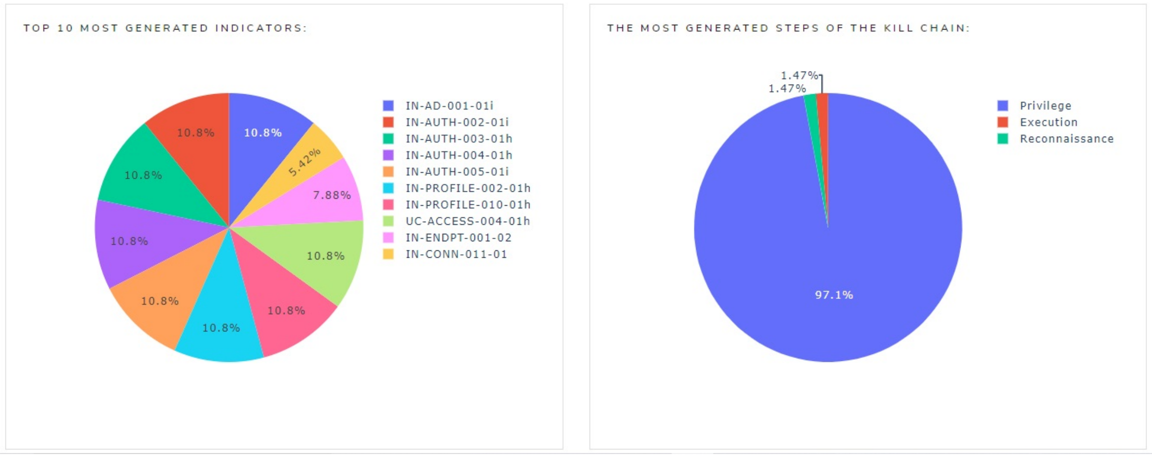


Figure 6.14 Most generated IoCs and kill chain steps by user from 2020-01-01 to 2021-07-06

After analyzing the first five users, it is time to investigate the sixth one who is actually the aforementioned user and has appeared among the top 10 users. By clicking on the related bar the user behavior in terms of kill chain pattern will be demonstrated. We will increase the date window to analyze any suspicious activity in the past (Figure 6.15). The dates are used in the filter section of query VQ4.1.

The analyst can also easily look at the behavior of the user in previous months to investigate the kill chain pattern of the user. As marked in Figure 6.16, there is a kill chain pattern in February (The exact date and other information can appear by mouse hovering).

Let us set the marked kill chain activity aside (as it is obviously flagged) and see if there are more malicious behaviors not detected as kill chain activities. In order to do so, we take a look at the IoCs generated by this user in the time window he was flagged as a high exfiltration risk user (2020-05-25 - 2021-05-26). As demonstrated in Figure 6.17, there are

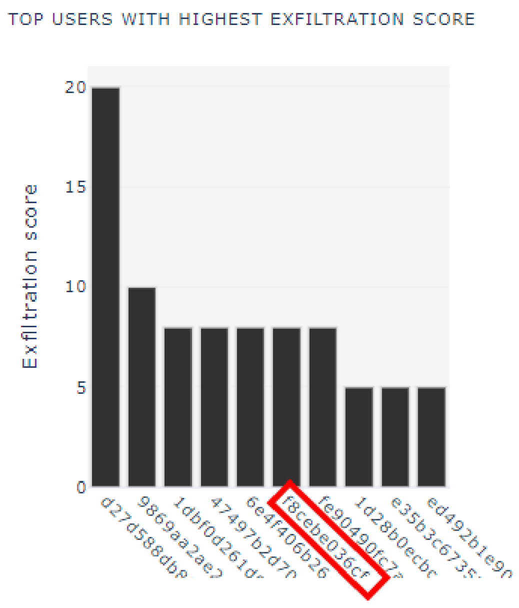


Figure 6.15 Users with highest exfiltration score from 2020-05-25 to 2021-05-26

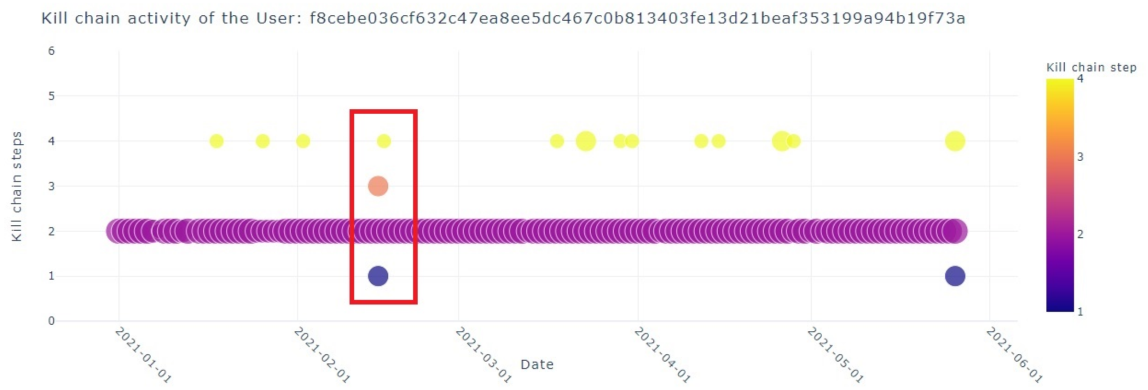


Figure 6.16 Most generated IoCs and kill chain steps by user from 2020-01-01 to 2021-07-06

different privilege IoCs in the list described in Table 6.5. These reports are visualized results of queries VQ2 and VQ3.

By a glance at Figure 6.17, the analyst understands most of the generated indicators are related to Privilege step. The description of these IoCs are summarized in Table 6.4. However, we know the majority of IoCs in this category are non-behavioral. Therefore, according to the fact that unusual activities are usually related to behavioral IoCs, we use the indicator type filter to see the statistics considering only behavioral IoCs. The results are shown in Figure 6.18 as a visualization of query VQ2 and VQ3 considering only :BehavioralIndicatorEvent

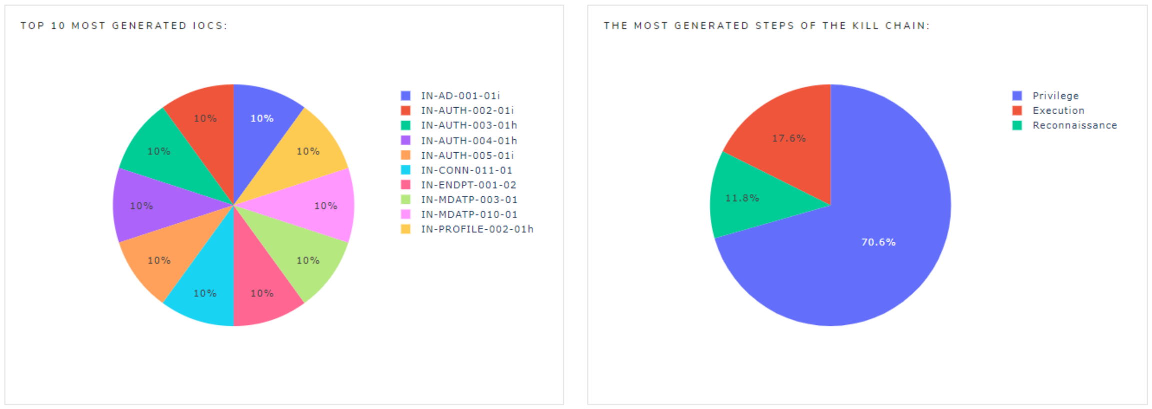


Figure 6.17 Most generated IoCs and kill chain steps by user from 2020-05-25 to 2021-05-26

class.

Table 6.4 Description of IoCs generated by the suspicious user in scenario no.4

Index	IoC description
1	Active Directory mouvement
2	Does not have an IT High Privilege Account (ex. N2)
3	Has at least one Account with High Business Permission rights
4	Had VPN access (or more)
5	Does not have an exemption to access the USB port
6	Miscellaneous administrator - History

By looking at the list of IoCs in Figure 6.18 the analyst who is familiar with IoC IDs, could quickly determine a malicious behavior. We bring the description of these IoCs in Table 6.5 so that what the user has done would be more clear.

As we can see in Table 6.5, the user has accessed a sensitive file and downloaded it. Also, he has sent an email to an address that appears to be his external email address. This email contains a file including confidential information. Obviously the behavior of this user should be investigated to see the files he has attached to the email and check whether he has access to these confidential data or not.

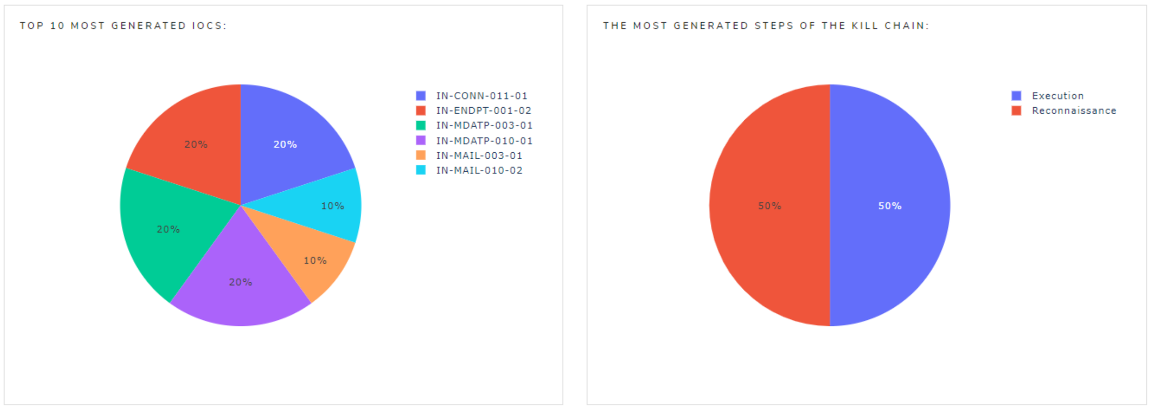


Figure 6.18 Most generated behavioral IoCs and kill chain steps by user from 2020-05-25 to 2021-05-26

Table 6.5 Description of behavioral IoCs generated by the suspicious user in scenario no.4

Index	IoC description
1	Modification of a sensitive file by a user.
2	Creation (download) of a sensitive file by a user.
3	Email sent externally (with attachment) to a recipient that appears to be their personal address
4	Email sent externally with a file attached

It also should be mentioned that although the user is following kill chain steps, we could not see the "Accumulation" step in Figure 6.16. The reason is one of the indicators that should have been categorized as "Accumulation" but is not categorized correctly. Despite this error, we could successfully detect the potentially malicious behavior of this user.

Result

To summarize, the malicious user is correctly flagged and the IVIIO user can detect a behavior deemed as threat to information security by looking at different information shown in the dashboard very quickly. It should be considered that we do not have access to the investigation level data and cannot move further than flagging the username. However, the user was successfully identified as a high risk user in IVIIO.

6.3.5 Scenario no.5

The last scenario is an observed scenario in which a normal user was flagged as a suspicious one. After investigation process the case was reported as a false positive.

Description

In this scenario we received 2528 IoCs generated by a user (with anonymized username "`d0eb86d62daa3e565b0a04d76e564004c5e5cc5cd82cc2fc34abf2e2b48ca4ef`") who was detected as a suspicious user and after investigation, they found out that he was a normal user. Unfortunately, we do not know the reason this user is flagged and investigated. The summary of the received file is listed below:

- Number of indicator events: 2528
- Number of distinct indicators: 44
- Number of distinct dates: 430
- Number of distinct users: 1

In this scenario, we try to see if the dashboard flags this user as a false positive or not. In another case, if we do not mark this user as a potential malicious user, we are still in the right path.

We have the activity of this user for about a year. Obviously, checking the dashboard for every day in a year to see if the user is reported or not, is not possible. To address this challenge, we take a look at the whole user behavior (with the help of VQ6) and check only the dates in which user has generated many indicators and is probably reported as a suspicious user. We know if the user is not reported in dates he has produced the highest amount of indicators, with high confidence we can say he is not reported during the whole time window.

As shown in Figure 6.19, there are three peak points in the user behavior on 2020-12-22, 2021-02-08 and 2021-04-11. We will look at the high risk users in these three dates to see if the aforementioned user is reported or not.

Investigation

In this section we will describe the process taken to see whether the user is reported by IVIIO or not. If he is not reported, there is no other way we could detect him as a high risk user

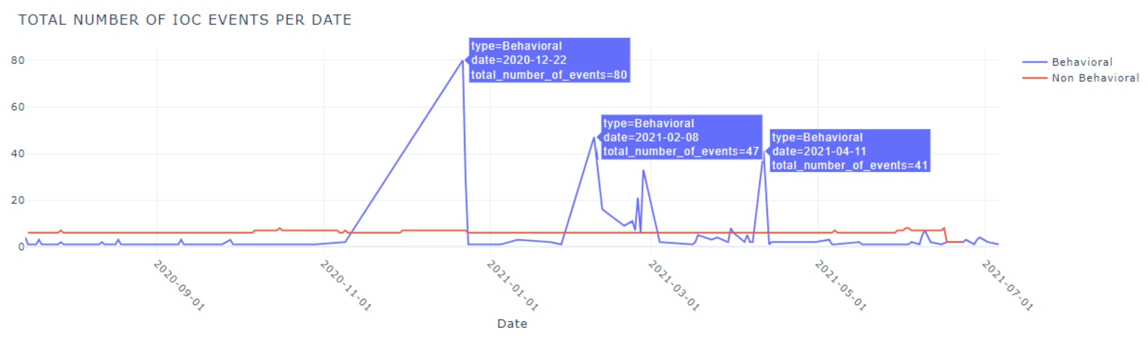


Figure 6.19 User activity from 2020-07-15 to 2021-07-06

and investigate his behavior.

Statistics on date 2020-12-22: Figure 6.20 summarizes the high risk users reported on 2020-12-22. These reports are generated based on the results of queries VQ1, VQ4 and VQ8. Despite the high score user "`d0eb86d62daa3e565b0a04d76e564004c5e5cc5cd82cc2fc34ab f2e2b48ca4ef`" had on this date, he is not mentioned among suspicious users.

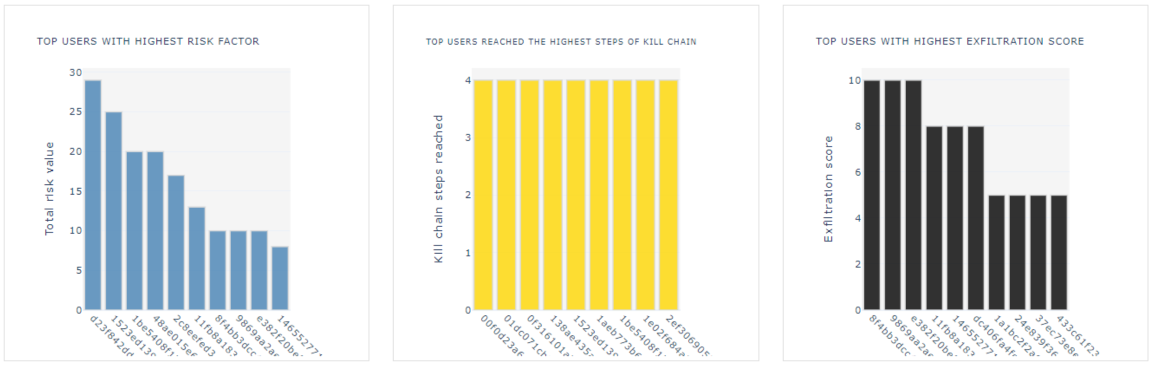


Figure 6.20 High risk users reported on 2020-12-22

Statistics on date 2021-02-08: The high risk users reported on 2020-12-22 are shown in Figure 6.21. Again, in this day, the aforementioned user is not reported.

Statistics on date 2021-04-11: According to Figure 6.22, although there are only few high risk users on this date, no report has listed the name of user we are looking for.

Result

Since the user is not reported among high risk users on the days he has generated many IoCs, we can estimate that he is never reported. Therefore, IVIIO performs correctly by not flagging the user who was wrongly detected by the currently used solution.

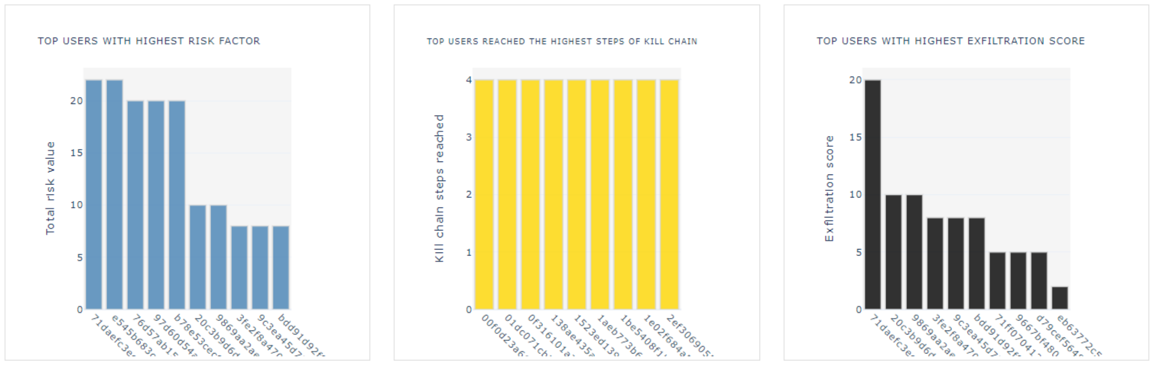


Figure 6.21 High risk users reported on 2021-02-08

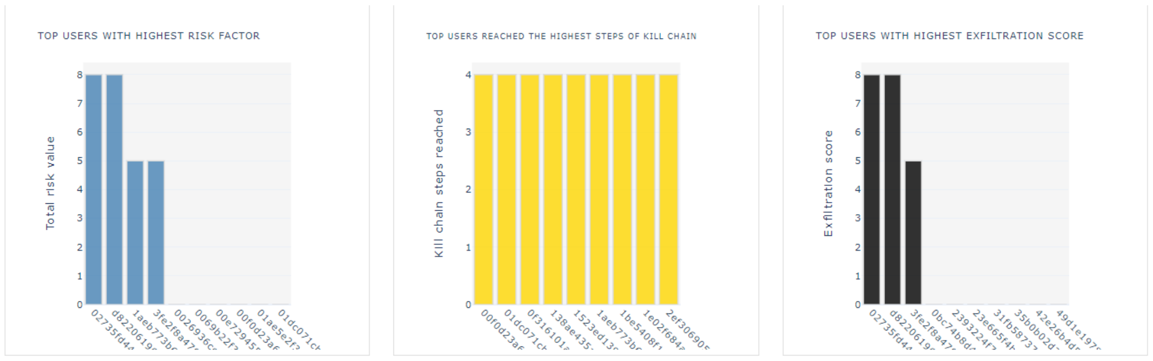


Figure 6.22 High risk users reported on 2021-04-11

6.4 Limitation

During the evaluation phase of this project, we faced some limitations that are summarized in the following paragraphs.

First, we did not always have enough information to tag IoCs as behavioral or non-behavioral. In these cases, we chose the most probable class according to the sensor information, similar IoCs and related kill chain step.

Furthermore, in all the cases investigated, more detailed information is required for making the right decision. For example, there might be some high privilege users who need to edit or send sensitive files to specific addresses. The IoCs generated by these users could be marked as false positive if we could correlate them with the requirements of user assigned tasks, user role, etc.

We believe the aforementioned limitations could be addressed in future works where we have access to contextual information and details of the IoC events. In this way, by modeling each domain and integrating the designed ontologies, every instance can be connected to all the required details. Also, different rules could be written to correlate the related information and make a definitive conclusion. Ontological modeling could facilitate reaching this goal since integrating different ontologies is a less challenging task in comparison with integrating different relational databases. To exemplify, there are many constraints (such as primary keys, foreign keys, null rules,etc.) that may prevent a database tuple to be added in the database. However, ontologies do this job by reasoning and provide a restriction-free framework [17]. This makes ontologies more suitable for integration purposes.

6.5 Summary

In this chapter, we discussed five scenarios to evaluate how IVIIO is working. We considered some assumptions including:

- All the information are collected.
- In a kill chain attack, an ordered sequence of kill chain steps from the first step to the last one is taken respectively.
- The reports in IVIIO are investigated daily.
- Each user is working in a team.

The first three scenarios are designed by a labmate and a research assistant and the other two are observed use cases in Desjardins. For each scenario, we will add the new events in the ontological database and check the dashboard to see how IVIIO can enhance the investigation process by automating queries and visualizing the results. The summary of the evaluation process and the results are provided in Table 6.6.

Table 6.6 Summary of scenarios evaluation

Scenario	Reported by IVIIO?	Real decision	IVIIO decision	Result
1	yes	TP	TP	Successful
2	partially	FP	FP and TP	Partially successful
3	yes	TP	TP	Successful
4	yes	TP	TP	Successful
5	no	FP	TN	Successful

The story of the first scenario is about a malicious user who is taking all the steps of the kill chain. We detected the suspicious user because he was listed in two reports: top users with highest risk factor and top users with highest exfiltration score. After finding the user, a kill chain activity resulted from many indicators showing sensitive data access and exfiltration was discovered. The user is flagged as a suspicious user so that his detailed behavior be investigated. Since this scenario is designed to be a kill chain attack, we can say IVIIO could successfully detect the intended activity and help to list the prominent IoCs to demonstrate what user has done as detailed as possible. However, as we do not have access to contextual information, we cannot go through the details of user roles, privileges and other helpful data to make a strict decision.

The second scenario was simulating a normal user who may be assumed as a suspicious user because of his high rate of IoC events. This user appeared among top users with highest risk factor and top users reached to the highest step of the kill chain. Looking at the kill chain activity of the user, a possible kill chain attack is detected. To retrieve more information, we decided to analyze the list of indicators generated by user on the dates he has taken the fourth step of the kill chain which is execution (data exfiltration). According to the investigation of the data related to six different dates, the user behavior was not likely malicious in general as he had the required privileges for the actions. However, the user behavior is flagged to be investigated in two specific dates in which he may not have the required privilege for using the USB port. The final result of the investigation could be acquired by considering more detailed information such as user privileges and role as well as details of the files copied to USB device.

The third scenario was supposed to show a potential malicious user who is generating random high risk indicators. Therefore, we faced a huge number of non-behavioral indicators that are normally not considered as important as behavioral ones. However, due to the large number of indicators, the user was ranked among top users with highest risk factor and top users with highest exfiltration score. Investigation process revealed many privilege changes. Hence, although user was authorized to use different channels (USB, printer, etc.), he flagged as a suspicious user to be investigated because he might steal or misuse credentials. We could conclude more accurately if we could have seen user role and compare the role with the acquired privileges.

Fourth scenario was related to an observed case in Desjardins. By looking at the diagram of total number of indicators per date for the related user, we found the dates in which the user might be listed in high risk users reports. Therefore, we select a date and demonstrate the user is listed as a user with high exfiltration score. In the investigation phase, we found a kill chain activity happened a few months ago. Separating behavioral and non-behavioral indicator analysis, we found the user is not authorized to have a high privilege account while he modified sensitive files and exfiltrate them using his personal email address. The real story was about a user modifying confidential files and exfiltrating them.

Finally, the last scenario is about a false positive case investigated in Desjardins. We looked for a report listing the related user on dates he has generated the highest number of indicators but could not see his name among high risk users. Therefore, it is very unlikely for this user to be reported on other dates.

To put the story in a nutshell, it is not possible to explicitly label the users as attackers or normal users based on current information. To do so, we need to have access to contextual information such as user role, privileges, tasks and details of the actions as well as psychological situation, employment information, etc. Therefore, we designed IVIIO to use as much information as possible to list the most high risk users and their activities to be investigated. In the future, our results could be improved by having access to contextual data in different domains, developing an ontological model for each domain and finally connecting the designed ontologies to reach a comprehensive model.

CHAPTER 7 CONCLUSION

Following the description and evaluation of IVIIO, in this chapter we will provide a summary of the work, discuss the research questions and provide a conclusion considering all the restrictions and limitations.

7.1 Summary of the work

In this research, a use case from the “Information security investigation” team in Desjardins financial institution is discussed. According to our meetings with experts, the analysts face a huge number of insider IoC events and have to investigate which ones are real threats and which ones are false positive events. For this purpose, they may need to query the database about target asset sensitivity, user’s behavior and other detailed information. This process is time-consuming and in some cases, complicated queries should be written to get the required information. Also, considering the whole insider threat monitoring system, according to the fact that the IoCs are coming from different security solutions, there will be different levels of granularity.

We developed an ontological model of insider IoCs and visualized the required reports in a tool named IVIIO to enhance the investigation process by providing summarized reports and automating part of the job. We also considered the Desjardins insider threat kill chain to analyze user behaviors and detect kill chain attacks.

In chapter 2, we summarized basic cyber security concepts in insider threat, IoC and cyber kill chain domains. Then we reviewed the state of the art of four security domains (insider threat, IoC, cyber kill chain and insider threat visualization) to be cognizant of the state of the art. Reviewing the literature, we figured out insider threats are more complicated to be detected than outsider ones since the attackers have already passed some of the security defense layers and are sometimes authorized to access sensitive information and assets. We noticed that using IoCs can provide a uniform structured data format of different indicators which results in improving indicator responding and management tasks. Studying different cyber kill chains demonstrated how unknown attacks can be detected by investigating malicious chain of behaviors. Finally, reviewing insider threat visualization tools, we noticed how effective visualization can be in security domain and insider threat detection and investigation.

Chapter 3 provided basic concepts and fundamental knowledge of ontology and semantic web and reviewed the related work in ontology design methodology and ontology usage in insider threat areas. After assessing different ontology design methodologies, we decided to use ATOM for designing our ontological model since it is a methodology to model cyber-security systems. Also, we were willing to see how ATOM works in a real cyber security environment. Furthermore, we explored the previous works trying to use ontologies in insider threat domain. We learned ontologies can be helpful in insider threat domain because of their logic-based nature, reusability, extendability, flexibility, integrability and sharing facilities. However, to the best of our knowledge, no ontology was designed to model kill chain attacks.

In chapter 4, we explained the data and environment specifications such as IoCs, IoC events, data sources, kill chain steps and so on. To get more insight on data, some general statistics of observed logs were provided.

Chapter 5 described the proposed solution named as IVIIO by providing the architecture, design and implementation processes related to the two main modules in IVIIO: ontological solution and visualization tool. We described how we deployed ATOM methodology to design our ontological model. To develop and populate this model, we used Protégé, GraphDB and related python libraries. Then, we moved to the next module which is visualization tool and explained the steps taken for designing the interactive dashboard of IVIIO. We also described every report shown in the dashboard, how they should be interpreted and how they change based on user interaction.

In order to evaluate IVIIO, we used three synthetic and two observed scenarios in chapter 6 and demonstrated how the solution can help the analyst to investigate the threat. We noticed that in some cases, there are enough information to flag an activity as a potential threat while for some other ones we need additional contextual information to make a decision.

7.2 Discussion

The main objective of this research was to develop an ontology for insider IoCs. To enhance the insider threat investigation process, we decided to detect kill chain attacks and visualize the reports. Therefore, we divided our objective into two research questions:

RQ1: What improvements can we make on the insider IoC investigation process using ontologies?

In the investigated scenarios we have seen situations in which we could benefit from ontological database capabilities. Ontology allows us to be able to find some entities by just taking their specific properties. In other words, if we are not sure about the type of an entity, we can call it using its specific properties: for example, assuming only behavioral indicators have the property x. When we query the behavioral IoCs generated by a user, there might be some

IoCs not explicitly labeled as behavioral indicators. In these cases, we can ignore the type and query all the entities with specific property *x*, to have all behavioral indicators whether they are labeled correctly or not. To make sure the result is not confused with other classes, we can relate the entity to be an instance of one of its super classes. In this example, we know the entity belongs to the class :Indicator or one of its sub classes. Therefore, we query all the instances of the class :Indicator that have the property *x*. According to Figure 7.1, if we do not know the indicator "ind_1" is an instance of the class ":BehavioralIndicator or :NonBehavioralIndicator, we can access it using the super class ":Indicator". This option is more useful when the sub classes of a class have different properties. Another example would be the cases where we have access to heterogeneous data coming from different sensors. Ontology could be used to see all the events as the same entity while each category have their own specific features and attributes. In future versions of the project that include more data layers and specific properties for each class, we can highlight this advantage more clearly since it allows to cover missing information and human errors that happened during the labeling process.

According to the research in the related field, ontologies are believed to have benefits in insider threat domain [101]. Reviewing related works mentioned in section 3.2.2, there are many advantages of using ontologies in insider threat domain including:

1. Allowing for logic-based queries
2. Avoiding having thousands of entries of redundant information
3. Facilitating sharing expert knowledge with research/operational communities
4. Facilitating reuse of the domain knowledge
5. Facilitating use of a consistent conceptual framework across different departments within an organization by sharing common understanding of the structure of information among people or software agents
6. Being integrated with other ontologies by using a formal representation of domain knowledge
7. Being extendable and customizable to fit specific needs and missions
8. Providing aid for evaluating an organization's Insider threat assessment profile

In addition, our literature review demonstrates different ontological databases could be integrated for high level correlation purposes. Therefore, future works focusing on modeling

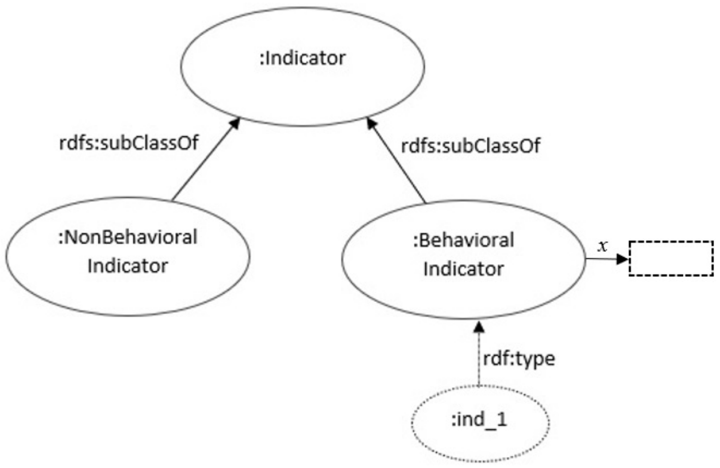


Figure 7.1 :ind_1 is defined to be an instance of the class :BehavioralIndicator and is inferred to be an instance of the class :Indicator.

other domains and contextual data could be integrated with our work to provide stronger investigations. Ontologies are known to be integrated and merged in a less restricted way in comparison with relational databases [17]. This can be counted as the other advantages of using ontologies.

According to our evaluation process, by keeping records of users activities in ontological database and updating them frequently, we can detect high risk users in any time window. In order to not being biased on a single approach, we report the users based on three different points of view: highest risk factor, highest exfiltreation score and highest kill chain step reached. Although we did not have the chance to run the solution in the real environment and provide the exact detection statistics, according to our evaluation scenarios, we estimate that if the solution was used every day, the chance of missing an attacker is quite small since we consider different high risk users. As demonstrated in evaluation scenarios, although there might be normal users reported as high risk users, we did not have any false negative result.

Finally, during the evaluation phase, we demonstrated the ability of our model to answer all of the desired questions and to detect suspicious users and their detailed activities. The analysts could change the variables inside SPARQL queries and make new ones to monitor activities of intended users in different time windows. However, in some complicated scenarios where additional contextual information were needed to make a decision, we cannot be sure whether an activity is related to an attack or not. We believe enriching the ontology with contextual information can solve this problem.

RQ2: How visualizing IoCs affects the insider threat investigation?

Although a significant amount of research is done on visualization in the security area, little work has been focused on visualizing insider threat use cases. Here, we mention how visualization affects the insider threat investigation. In this research, we tried to develop a visualization tool connected to an ontological database of insider IoCs to provide insightful reports to the security analysts without forcing them to have any ontological knowledge and experience while benefiting from ontological model advantages. IVIIO allows the analysts to see the result of their desired query by selecting filters and reports instead of writing complicated queries.

In the evaluation phase, we demonstrated how visualization removes the problem of learning a new tool and language and decreases the time required for querying the database and interpreting complicated reports. Working with IVIIO, there was no need to learn how to write long complicated SPARQL queries, interpreting the outputs and working with enormous amount of data. Analysts can make their desired queries using the provided filters or by interacting with the reports. It goes beyond doubt that in security operations where every second plays a pivotal role, automating parts of the job and increasing threat response speed is very valuable.

Furthermore, we noticed that deploying visualized tools can decrease the false positive rate. Visualizing data enhances detecting trends and exceptions even in the most complex scenarios [110]. Based on experimental observations, this goal cannot be met using machine learning, rule based or other proposed approaches successfully as every threat can be different from the previous known ones and using these methods may result in high rate of false positives. For this reason, instead of using fully automated methods, many organizations decided to hire a team of security analysts to analyze the data and make the final decisions by interpreting reports and incident IoCs. In this research we showed how different reports of a user (such as kill chain activity, comparing user behavior with his teammates, top generated indicators and kill chain steps, etc.) are combined to increase reliability and help the analyst to decide whether there is a real threat or it is part of user's regular tasks.

Additionally, visualization provides human understandable information acquired from huge amount of data. In our evaluation process, we demonstrated IVIIO can be used to detect a small list of suspicious users to be investigated. Among all the users who are generating different indicators, the ones with the highest risk factor, highest exfiltreation score and highest kill chain step reached, are listed in the dashboard. For example, the analysts can also see a clear track of kill chain steps taken by the user and detect kill chain activities in a glance. Certainly, it takes much more time to analyze non-visualized lists of these activities.

Last but not least, we learned insider threat investigation process is a quite complex task that varies case by case and day by day. Therefore, we still need security analysts to have explicit results. However, we can accelerate the process by automating parts of the job. In this research, we have automated parts of the insider IoC investigation process including insider IoC monitoring, querying the database, analyzing users' activities, detecting kill chain attacks and reporting high risk users. We demonstrated the solution is effective to find potential malicious users and could help to investigate insider threats as long as the available information in the dashboard is enough. However, in some cases we need additional contextual information (such as user role, privileges, tasks, psychological profile, etc.) for making the final decision. Due to the fact that in this project we did not have access to these kinds of data, we hope future works can address these types of challenges since our work provides a basic model to be extended and integrated with other models for having a more robust insider threat investigation assistant tool.

7.3 Limitations

During this experiment, we were provided with the required information but there were some restrictions that did not allow us to access all the desired data and have deeper ontological model. We also were eager to add contextual and psychological data into our model to evaluate the correlation between technical and non-technical indicators. However, these kinds of data were not available at the moment. Therefore, we tried to model everything we had in a structured way such that future works could extend our model easily.

In addition, in some cases such as categorizing behavioral and non-behavioral indicators or defining user teams, we did not have enough information to clearly describe the data. For example, the description field of the indicator table was sometimes empty and the ID was not clearly explaining the indicator. Also, for some users, we did not have the information about their teams. We addressed these challenges by mapping entities into the most likely category. In case of lack of indicator explanation, we decided based on the kill chain step and indicator type of the similar indicators (the ones coming from the same sensor and with proximate risk factors). In case of lack of user team information, we defined 1-member teams for each user.

Another limitation to be mentioned is not being able to have the model evaluated by the analysts working in the field. Our ideal evaluation process was launching IVIIO at Desjardins and asking analysts to work with it to see how they find it in comparison with the traditional routine they had before. However, due to restrictions, we had to use labeled data and evaluate the solution offine.

Furthermore, we assumed we receive all the IoCs related to user behavior and they are all correctly categorized to be related to a step of the kill chain. In reality, this assumption is not always correct. As an example we can refer to scenario no.4 in chapter 6 in which a wrongly categorized indicator prevented the solution to detect a kill chain activity.

Finally, we only considered a kill chain activity if the user takes all the steps without skipping one in the middle. This means the user may jump back to the previous steps but cannot skip one while moving upward (Figure 7.2). In real world, there may be kill chain activities not containing all the steps.

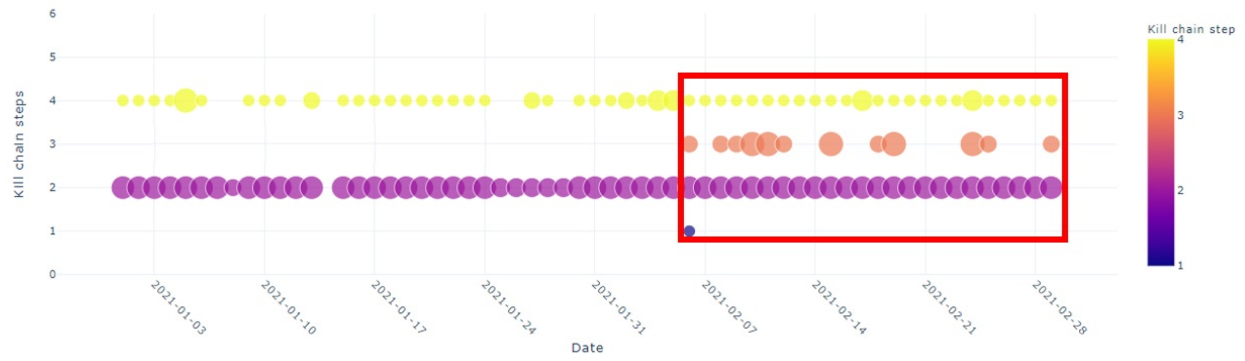


Figure 7.2 Kill chain activity is detected only when no middle step is missed.

7.4 Future work

Here, we will address limitations mentioned in the last section as future works to expand the project and cover more data and attack scenarios.

First, we suggest to study a boarder part of the database and expand the ontological model to cover more abstract layers of data to benefit from inferences. Providing this data can also help to detect more complicated types of attacks and suspicious users. Also, correlating technical and non-technical indicators such as sociological or psychological ones can result in more reliable detection.

Additionally, different future researches could focus on modeling different domains. The ontologies produced by all the teams can be integrated to provide a comprehensive ontological model that is able to correlate relevant data and make strong decisions.

Moreover, missing and non-integrated data is an important issue to be addressed in future works. In these cases, abductive reasoning can be deployed to generate hypotheses about the entities with incomplete information. Also, there could be meetings with the experts for cleaning data where needed. Finally, we suggest to have a centralized shared data source available to the researchers and employees, so that no one may work on an expired version of a file.

Finally, the best way of evaluating these solutions is being used in the real environment by the same people who are using other solutions for a reasonable period of time and getting their valuable feedback in different areas (such as completeness of the solution, flexibility, usability, speed, soundness, etc.). The information collecting method could be chosen from observation, interview, survey, etc. based on the restrictions and availability of the analysts. There are also different methods for evaluating the designed ontologies that can be used by future researchers to evaluate the ontological model.

REFERENCES

[1] Gartner Inc. Gartner PeerInsight. [Accessed March 3, 2021]. [Online]. Available: https://www.gartner.com/reviews/home

[2] L. Liu, O. De Vel, Q.-L. Han, J. Zhang, and Y. Xiang, "Detecting and preventing cyber insider threats: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1397–1417, 2018.

[3] E. M. Hutchins, M. J. Cloppert, R. M. Amin *et al.*, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues in Information Warfare & Security Research*, vol. 1, no. 1, p. 80, 2011.

[4] FireEye. Mandiant apt1 report: Exposing one of china's cyber espionage units. [Accessed May 19, 2021]. [Online]. Available: https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf

[5] B. D. Bryant and H. Saiedian, "A novel kill-chain framework for remote security log analysis with siem software," *computers & security*, vol. 67, pp. 198–210, 2017.

[6] ——, "Improving siem alert metadata aggregation with a novel kill-chain based classification model," *Computers & Security*, vol. 94, p. 101817, 2020.

[7] B. Balakrishnan *et al.*, "Security data visualization," *SANS Institute InfoSec Reading Room*, 2015.

[8] R. Marty, *Applied security visualization.* Addison-Wesley Professional, 2008.

[9] "The cost of malicious cyber activity to the u.s. economy," The Council of Economic Advisers, Tech. Rep., 2018, [Accessed Jun 10, 2020]. [Online]. Available: https://www.whitehouse.gov/wp-content/uploads/2018/03/The-Cost-of-Malicious-Cyber-Activity-to-the-U.S.-Economy.pdf

[10] H. Schulze, "Insider threat report," Fortinet Cybersecurity Insiders, Tech. Rep., 2019, [Accessed Mar 04, 2020]. [Online]. Available: https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/insider-threat-report.pdf

[11] E. E. Schultz, "A framework for understanding and predicting insider attacks," *Computers & Security*, vol. 21, no. 6, pp. 526–531, 2002.

[12] K. Nance and R. Marty, "Identifying and visualizing the malicious insider threat using bipartite graphs," in *2011 44th Hawaii International Conference on System Sciences*. IEEE, 2011, pp. 1–9.

[13] D. L. Costa, M. J. Albrethsen, and M. L. Collins, "Insider threat indicator ontology," Carnegie Mellon University, Pittsburgh, PA, United States, Tech. Rep., 2016.

[14] G. Kul and S. Upadhyaya, "A preliminary cyber ontology for insider threats in the financial sector," in *Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats*, 2015, pp. 75–78.

[15] Quick facts about desjardins. [Accessed Oct 12, 2021]. [Online]. Available: https://www.desjardins.com/ca/about-us/desjardins/who-we-are/quick-facts/index.jsp

[16] L. F. Dias and M. Correia, "Big data analytics for intrusion detection: an overview," in *Handbook of Research on Machine and Deep Learning Applications for Cyber Security*. IGI Global, 2020, pp. 292–316.

[17] C. Martinez-Cruz, I. J. Blanco, and M. A. Vila, "Ontologies versus relational databases: are they so different? a comparison," *Artificial Intelligence Review*, vol. 38, no. 4, pp. 271–290, 2012.

[18] I. Kotenko, O. Polubelova, and I. Saenko, "The ontological approach for siem data repository implementation," in *2012 IEEE International Conference on Green Computing and Communications*. IEEE, 2012, pp. 761–766.

[19] C. Islam, M. A. Babar, and S. Nepal, "An ontology-driven approach to automating the process of integrating security software systems," in *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*. IEEE, 2019, pp. 54–63.

[20] A. Sadighian, "Intrusion detection from heterogeneous sensors," PhD thesis, Dép. de génie informatique, École Polytechnique de Montréal, Montréal, QC, 2015. [Online]. Available: https://publications.polymtl.ca/1702/

[21] S. Malenfant-Corriveau, "Proposition d'une méthode de développement d'ontologie pour un système expert en sécurité," Master's thesis, Dép. de génie électrique, École Polytechnique de Montréal, Montréal, QC, 2017. [Online]. Available: https://publications.polymtl.ca/2922/

[22] T. DUCHARME, "D'Étection d'intrusion À l'aide d'un systÈme expert basÉ sur l'ontologie," Master's thesis, Dép. de génie informatique, École Polytechnique de

Montréal, Montréal, QC, 2017. [Online]. Available: https://publications.polymtl.ca/2923/

[23] R. Chinchani, D. Ha, A. Iyer, H. Q. Ngo, and S. Upadhyaya, “Insider threat assessment: Model, analysis and tool,” in *Network security.* Springer, 2010, pp. 143–174.

[24] S. L. Pfleeger, J. B. Predd, J. Hunker, and C. Bulford, “Insiders behaving badly: Addressing bad actors and their actions,” *IEEE transactions on information forensics and security*, vol. 5, no. 1, pp. 169–179, 2009.

[25] Q. Althebyan and B. Panda, “A knowledge-base model for insider threat prediction,” in *2007 IEEE SMC Information Assurance and Security Workshop.* IEEE, 2007, pp. 239–246.

[26] C. W. Probst, J. Hunker, M. Bishop, and D. Gollmann, “08302 summary–countering insider threats,” in *Dagstuhl Seminar Proceedings.* Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2008.

[27] F. L. Greitzer, A. P. Moore, D. M. Cappelli, D. H. Andrews, L. A. Carroll, and T. D. Hull, “Combating the insider cyber threat,” *IEEE Security & Privacy*, vol. 6, no. 1, pp. 61–64, 2008.

[28] M. Bishop, “Position: " insider" is relative,” in *Proceedings of the 2005 workshop on New security paradigms*, 2005, pp. 77–78.

[29] J. Predd, S. L. Pfleeger, J. Hunker, and C. Bulford, “Insiders behaving badly,” *IEEE Security & Privacy*, vol. 6, no. 4, pp. 66–70, 2008.

[30] G. J. Silowash, D. M. Cappelli, A. P. Moore, R. F. Trzeciak, T. Shimeall, and L. Flynn, “Common sense guide to mitigating insider threats,” 2012.

[31] A. Kim, J. Oh, J. Ryu, J. Lee, K. Kwon, and K. Lee, “Sok: A systematic review of insider threat detection.” *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 10, no. 4, pp. 46–67, 2019.

[32] F. L. Greitzer, D. Frincke, and M. Zabriskie, “Social/ethical issues in predictive insider threat monitoring,” in *Information assurance and security ethics in complex systems: Interdisciplinary perspectives.* IGI Global, 2011, pp. 132–161.

[33] M. Theoharidou, S. Kokolakis, M. Karyda, and E. Kiountouzis, “The insider threat to information systems and the effectiveness of iso17799,” *Computers & Security*, vol. 24, no. 6, pp. 472–484, 2005.

[34] J. Hunker and C. W. Probst, "Insiders and insider threats-an overview of definitions and mitigation techniques." *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 2, no. 1, pp. 4–27, 2011.

[35] CERT Insider Threat Center, "Unintentional insider threats: A foundational study," *cahier de recherche CMU/SEI-2013-TN-022, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA*, vol. 18, 2013.

[36] B. Reed and J. Care, "How to build incident response scenarios for insider threats ID G00380185," Gartner Inc., Tech. Rep., 2019.

[37] P. Furtado, "Strategies for midsize enterprises to mitigatethe insider threat ID G00451251," Gartner Inc., Tech. Rep., 2020.

[38] McAfee. What Is DLP and How Does It Work? [Accessed July 23, 2021]. [Online]. Available: https://www.mcafee.com/enterprise/en-ca/security-awareness/data-protection/how-data-loss-prevention-dlp-technology-works.html

[39] Gartner Inc. Gartner Glossary - Endpoint Protection Platform (EPP). [Accessed March 3, 2021]. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/endpoint-protection-platform-epp.

[40] ——. Gartner Glossary - Identity and Access Management (IAM). [Accessed March 3, 2021]. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/identity-and-access-management-iam

[41] ——. Gartner Glossary - Mobile Device Management (MDM). [Accessed March 3, 2021]. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/mobile-device-management-mdm

[42] Microsoft. Secure access to resources with multifactor authentication. [Accessed Jul 05, 2021]. [Online]. Available: https://www.microsoft.com/en-ca/security/business/identity-access-management/mfa-multi-factor-authentication

[43] ——. Privileged Access Management for Active Directory Domain Services. [Accessed Jul 05, 2021]. [Online]. Available: https://docs.microsoft.com/en-us/microsoft-identity-manager/pam/privileged-identity-management-for-active-directory-domain-services

[44] FireEye. What is UEBA? Definition and Benefits. [Accessed Jul 05, 2021]. [Online]. Available: https://www.fireeye.com/products/helix/what-is-ueba.html

[45] CERT Insider Threat Center, "Insider threat control: Using a siem signature to detect potential precursors to it sabotage," *Software Engineering Institute, Carnegie Mellon University Carnegie Mellon University*, 2011.

[46] Solarwinds. Centralized Log Management. [Accessed Jul 05, 2021]. [Online]. Available: https://www.solarwinds.com/security-event-manager/use-cases/centralized-log-management

[47] Gartner Inc. Gartner Glossary -Security Information And Event Management (SIEM). [Accessed March 3, 2021]. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/security-information-and-event-management-siem

[48] McAfee. What Is a Security Operations Center (SOC)? [Accessed Jul 05, 2021]. [Online]. Available: https://www.mcafee.com/enterprise/en-ca/security-awareness/operations/what-is-soc.html

[49] Gartner Inc. Gartner Glossary - Security Orchestration, Automation and Response (SOAR). [Accessed March 3, 2021]. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/security-orchestration-automation-response-soar

[50] FireEye. Cyber Threat Intelligence 101. [Accessed Jul 05, 2021]. [Online]. Available: https://www.fireeye.com/mandiant/threat-intelligence/what-is-cyber-threat-intelligence.html

[51] C. Harrington, "Sharing indicators of compromise: An overview of standards and formats," *EMC Critical Incident Response Center*, 2013.

[52] D. Bianco, "The pyramid of pain," *Enterprise Detection & Response*, 2013.

[53] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," *Insider Attack and Cyber Security*, pp. 69–90, 2008.

[54] A. Azaria, A. Richardson, S. Kraus, and V. Subrahmanian, "Behavioral analysis of insider threat: A survey and bootstrapped prediction in imbalanced data," *IEEE Transactions on Computational Social Systems*, vol. 1, no. 2, pp. 135–155, 2014.

[55] I. Homoliak, F. Toffalini, J. Guarnizo, Y. Elovici, and M. Ochoa, "Insight into insiders and it: A survey of insider threat taxonomies, analysis, modeling, and countermeasures," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–40, 2019.

[56] M. R. Randazzo, M. Keeney, E. Kowalski, D. Cappelli, and A. Moore, “Insider threat study: Illicit cyber activity in the banking and finance sector,” Carnegie Mellon University, Pittsburgh, PA, United States, Tech. Rep., 2005.

[57] MITRE. Common Vulnerabilities and Exposures. [Accessed May 21, 2021]. [Online]. Available: https://cve.mitre.org/

[58] ——. Common Platform Enumeration. [Accessed May 21, 2021]. [Online]. Available: https://cpe.mitre.org/

[59] ——. Common Weakness Enumeration. [Accessed May 21, 2021]. [Online]. Available: https://cwe.mitre.org/

[60] ——. Common Attack Pattern Enumeration and Classification. [Accessed May 21, 2021]. [Online]. Available: https://capec.mitre.org/

[61] ——. Adversarial Tactics, Techniques, and Common Knowledge. [Accessed May 21, 2021]. [Online]. Available: https://attack.mitre.org/

[62] F. L. Greitzer, J. D. Lee, J. Purl, and A. K. Zaidi, “Design and implementation of a comprehensive insider threat ontology,” *Procedia Computer Science*, vol. 153, pp. 361–369, 2019.

[63] A. Duncan, S. Creese, and M. Goldsmith, “A combined attack-tree and kill-chain approach to designing attack-detection strategies for malicious insiders in cloud computing,” in *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE, 2019, pp. 1–9.

[64] P. Chen, L. Desmet, and C. Huygens, “A study on advanced persistent threats,” in *IFIP International Conference on Communications and Multimedia Security*. Springer, 2014, pp. 63–72.

[65] P. Reidy. (2013) Combating the insider threat at the fbi: Real world lessons learned. [Online]. Available: https://media.blackhat.com/us-13/US-13-Reidy-Combating-the-Insider-Threat-At-The-FBI-Slides.pdf

[66] Tripwire, Inc. The insider threat: detecting indicators of human compromise. [Accessed Jul 27, 2020. [Online]. Available: https://www.tripwire.com/misc/the-insider-threat-detecting-indicators-of-human-compromise-register

[67] F. Howarth. Evolving uses of the kill chain framework. [Accessed May 7, 2021]. [Online]. Available: https://www.intelligentcio.com/me/wp-content/uploads/sites/12/2018/03/LogRhythm_Evolving_uses_of_kill_chain_framework_tlm_bloor.pdf

[68] P. A. Legg, "Visualizing the insider threat: challenges and tools for identifying malicious user activity," in *2015 IEEE Symposium on Visualization for Cyber Security (VizSec)*. IEEE, 2015, pp. 1–7.

[69] B. Haim, E. Menahem, Y. Wolfsthal, and C. Meenan, "Visualizing insider threats: An effective interface for security analytics," in *Proceedings of the 22nd International Conference on Intelligent User Interfaces Companion*, 2017, pp. 39–42.

[70] C. Biemann, "Ontology learning from text: A survey of methods." in *LDV forum*, vol. 20, no. 2, 2005, pp. 75–93.

[71] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: principles and methods," *Data & knowledge engineering*, vol. 25, no. 1-2, pp. 161–197, 1998.

[72] N. Guarino, *Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy.* IOS press, 1998, vol. 46.

[73] C. Keet, "An introduction to ontology engineering." v1," 2018.

[74] I. Horrocks, P. F. Patel-Schneider, and F. Van Harmelen, "From shiq and rdf to owl: The making of a web ontology language," *Journal of web semantics*, vol. 1, no. 1, pp. 7–26, 2003.

[75] F. Baader, I. Horrocks, C. Lutz, and U. Sattler, *Introduction to description logic.* Cambridge University Press, 2017.

[76] F. Manola, E. Miller, B. McBride *et al.*, "Rdf primer," *W3C recommendation*, vol. 10, no. 1-107, p. 6, 2004.

[77] J. H. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubézy, H. Eriksson, N. F. Noy, and S. W. Tu, "The evolution of protégé: an environment for knowledge-based systems development," *International Journal of Human-computer studies*, vol. 58, no. 1, pp. 89–123, 2003.

[78] The Apache Software Foundation. (2021) Apache Jena. [Accessed Aug 22, 2021]. [Online]. Available: https://jena.apache.org/index.html

[79] A. Kalyanpur, B. Parsia, and J. Hendler, "A tool for working with web ontologies," *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 1, no. 1, pp. 36–49, 2005.

[80] R. H. Güting *et al.*, *GraphDB: a data model and query language for graphs in databases.* Citeseer, 1994.

[81] O. Erling, "Virtuoso, a hybrid RDBMS/graph column store," *IEEE Data Engineering Bulletin*, vol. 35, pp. 3–8, 2012.

[82] A. Farquhar, R. Fikes, and J. Rice, "The ontolingua server: A tool for collaborative ontology construction," *International journal of human-computer studies*, vol. 46, no. 6, pp. 707–727, 1997.

[83] Stardog. Stardog Union Inc. [Accessed Aug 22, 2021]. [Online]. Available: https://www.stardog.com/

[84] K. I. Kotis, G. A. Vouros, and D. Spiliotopoulos, "Ontology engineering methodologies for the evolution of living and reused ontologies: status, trends, findings and recommendations," *The Knowledge Engineering Review*, vol. 35, 2020.

[85] D. B. Lenat and R. V. Guha, *Building large knowledge-based systems; representation and inference in the Cyc project.* Addison-Wesley Longman Publishing Co., Inc., 1989.

[86] E. Hovy, "Comparing sets of semantic relations in ontologies," in *The semantics of relationships.* Springer, 2002, pp. 91–110.

[87] J. F. Sowa, "Building large knowledge-based systems: Representation and inference in the cyc project: Db lenat and rv guha," 1993.

[88] M. Grüninger and M. S. Fox, "Methodology for the design and evaluation of ontologies," 1995.

[89] M. Uschold and M. King, *Towards a methodology for building ontologies.* Citeseer, 1995.

[90] M. Uschold and M. Gruninger, "Ontologies: Principles, methods and applications," *The knowledge engineering review*, vol. 11, no. 2, pp. 93–136, 1996.

[91] A. Bernaras, "Building and reusing ontologies for electrical network applications," *Proc. of ECAI 96, 1996*, pp. 298–302, 1996.

[92] B. Swartout, R. Patil, K. Knight, and T. Russ, "Toward distributed use of large-scale ontologies," in *Proc. of the Tenth Workshop on Knowledge Acquisition for Knowledge-Based Systems*, 1996, pp. 138–148.

[93] M. Fernandez, A. Gomez-Perez, and N. Juristo, "Methontology: from ontological art towards ontological engineering," in *Proceedings of the AAAI97 spring symposium series on ontological engineering.* Stanford, USA, 1997, pp. 33–40.

[94] D. J. Schultz *et al.*, "Ieee standard for developing software life cycle processes," *IEEE Std*, pp. 1074–1997, 1997.

[95] A. De Nicola, M. Missikoff, and R. Navigli, "A software engineering approach to ontology building," *Information systems*, vol. 34, no. 2, pp. 258–275, 2009.

[96] M. C. Suárez-Figueroa, "Neon methodology for building ontology networks: specification, scheduling and reuse," Ph.D. dissertation, Informatica, 2010.

[97] S. Peroni, "A simplified agile methodology for ontology development," in *OWL: Experiences and Directions–Reasoner Evaluation.* Springer, 2016, pp. 55–69.

[98] M. Poveda-Villalón, M. C. Suárez-Figueroa, and A. Gómez-Pérez, "Validating ontologies with oops!" in *International conference on knowledge engineering and knowledge management.* Springer, 2012, pp. 267–281.

[99] B. Aleman-Meza, P. Burns, M. Eavenson, D. Palaniswami, and A. Sheth, "An ontological approach to the document access problem of insider threat," in *International Conference on Intelligence and Security Informatics.* Springer, 2005, pp. 486–491.

[100] V. Raskin, J. M. Taylor, and C. F. Hempelmann, "Ontological semantic technology for detecting insider threat and social engineering," in *Proceedings of the 2010 New Security Paradigms Workshop*, 2010, pp. 115–128.

[101] D. L. Costa, M. L. Collins, S. J. Perl, M. J. Albrethsen, G. J. Silowash, and D. L. Spooner, "An ontology for insider threat indicators development and applications," Carnegie Mellon University, Pittsburgh, PA, United States, Tech. Rep., 2014.

[102] S. Barnum, "Standardizing cyber threat intelligence information with the structured threat information expression (stix)," *Mitre Corporation*, vol. 11, pp. 1–22, 2012.

[103] MITRE. (2014) Cyber Observable eXpression. [Accessed June 02, 2021]. [Online]. Available: http://cybox.mitre.org/language/version2.1/

[104] F. L. Greitzer, M. Imran, J. Purl, E. T. Axelrad, Y. M. Leong, D. Becker, K. B. Laskey, and P. J. Sticha, "Developing an ontology for individual and organizational sociotechnical indicators of insider threat risk." in *STIDS*, 2016, pp. 19–27.

[105] F. Greitzer, J. Purl, Y. M. Leong, and D. S. Becker, "Sofit: Sociotechnical and organizational factors for insider threat," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 197–206.

[106] V. Mavroeidis, K. Vishi, and A. Jøsang, "A framework for data-driven physical security and insider threat detection," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2018, pp. 1108–1115.

[107] What is power bi? [Accessed Sep 10, 2021]. [Online]. Available: https://powerbi.microsoft.com/en-us/what-is-power-bi/

[108] SPARQLWrapper 1.8.5. [Accessed Aug 24, 2020. [Online]. Available: https://pypi.org/project/SPARQLWrapper/

[109] Plotly. Dash Python User Guide. [Accessed Aug 13, 2020. [Online]. Available: https://dash.plotly.com/

[110] H. Shiravi, A. Shiravi, and A. A. Ghorbani, "A survey of visualization systems for network security," *IEEE Transactions on visualization and computer graphics*, vol. 18, no. 8, pp. 1313–1329, 2011.