Towards Collaborative Cyber Threat Intelligence for Security Management

Oleksii Osliak[1,2], Andrea Saracino[1], Fabio Martinelli[1] and Theo Dimitrakos[3]
[1]*Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Pisa, Italy*
[2]*Department of Computer Science, University of Pisa, Pisa, Italy*
[3]*Huawei Technologies, Munich, Germany*

Keywords: Access Control, Policy Update, Cyber Threat Intelligence, Amazon Web Services.

Abstract: Managing access to resources is one of the security mechanisms used for protecting the organization's assets from unauthorized usage, and thus potential data leaks. Thus, keeping access control policies up to date is a crucial task for any organization. However, the access control policy update process usually requires direct interaction of security specialists, which have knowledge and experience in counteracting abuse of privileges. Therefore, in this paper, we consider access control policies update using collaborative knowledge in the latest cyber activities. We describe the correlation between security policies and security reports using ontology for cybersecurity. Finally, we present a framework that enables access control policies update within the Cloud infrastructure offered by Amazon using Cyber Threat Intelligence.

1 INTRODUCTION

An important security requirement for any information management system is to protect data and resources from unauthorized access, and improper modifications still providing availability of those resources. An enforcement of such protection requires that every access to a system, resources and data is controlled only allowing accesses that should be authorized. The evaluation of a certain access request to execute an operation on a resource is done according to security policies, which define rules for access control regulation. Security policies can allow or deny the execution of certain operations on resourses depending on roles assigned to users or attributes that characterize a particular user. Additionally, security policies can contain other attributes used to describe context information (e.g., date, time, location, etc.) that can influence decision making. Since security policies play a critical role in organizations' assets protection, it is important to regularly update policies and control the way those policies are updated. The security policy update process may be caused by various factors including the data breach that took place in a system and external knowledge about potential data breaches. Nowadays, security specialists from various organizations and communities share threat-related information better known as *Cyber Threat Intelligence* (CTI) to prevent cyber-attacks and their spread. According to NIST (Johnson et al., 2016), CTI sharing allows developing new and improving existing security mechanisms to predict and prevent potential cyberattacks and defend organizations' assets. However, various tools that support CTI usage and enforcement, usually require direct human interaction which is a cumbersome process that might require a large amount of time. Moreover, security specialists share CTI using different formats, thus following different ontologies. In this paper, firstly we describe the interconnectivity between security reports and access control policies components using ontology. Secondly, we propose and describe a framework and its components for access control management within the Cloud environment using collaborative knowledge about cyberattacks in the form of structured reports. Finally, we present practical example of the usage of the proposed approach.

The paper is structured as follows. In Section 2, we provide simple formalization of *Attribute Based Access Control* model together with a definition of *Cyber Threat Intelligence* and tools used for its sharing. Section 3 describes the correlation between security reports and security policies. Section 4 presents and describes the security policy update process. Section 5 describes the proposed framework as well as its components. Finally, we provide related work in Sec-

Osliak, O., Saracino, A., Martinelli, F. and Dimitrakos, T.
Towards Collaborative Cyber Threat Intelligence for Security Management.
DOI: 10.5220/0010191403390346
In *Proceedings of the 7th International Conference on Information Systems Security and Privacy (ICISSP 2021)*, pages 339-346
ISBN: 978-989-758-491-6

tion 6, and Section 7 concludes the paper with a plan for future work.

2 BACKGROUND

In this section, we provide a background knowledge in access control and CTI approaches.

2.1 Access Control

Proper control of access to assets is one of the fundamental techniques for securing assets against cyberattacks and data abuse. Existing techniques and models like *Role Based Access Control* (RBAC) (Sandhu, 1998) and *Attribute Based Access Control* (ABAC) (Jin et al., 2012) are widely used by organizations for controlling access both to physical and cyber resources. Meanwhile, RBAC and ABAC are the most used models that follow different approaches. While subjects in the RBAC policies are defined through assigned roles (e.g., professor, student), in the ABAC model, both subjects and objects defined through the set of corresponding attributes. Thus, decision-making depends on the current attribute values. Furthermore, the ABAC policy allows defining context information through the set of corresponding attributes and considers roles assigned to subjects as additional attributes. Hence, the ABAC model allows policymakers to define fine-grained policies and express certain conditions through environmental attributes that do not belong directly to the subject or object.

In this work, we consider access control policies as *policy objects* that are specified by *subjects*, *objects*, *environment* and *operations*. Formally, we denote S, O, E and OP as finite sets of subjects, objects, environment and possible operations allowed in the system respectively. SA, OA and EA denote a set of subject, object and environmental attributes respectively. The policy object P is defined in a form of a tuple $P := \langle SA, OA, EA, op \rangle$, where SA, OA and EA denote subject, object and environmental attribute conditions, defined as sets of attribute name-value pairs. Finally, the $op \in OP$ is possible operation from a set of operations OP. The following example presents a policy that allows a subject with a role *student* or *supervisor* to execute *read* and *write* operations on a *file1* and *file2* iff the *current time* when the system received a request is in the range between 9 and 18.

$$
\begin{aligned}
P_i := \langle & \langle \exists role.student \lor role.supervisor \rangle \land \\
& \langle \exists operation.read \land operation.write \rangle \land \\
& \langle \exists object.file1 \land object.file2 \rangle \land \\
& \langle \exists currentTime \geq 9 \land currentTime \leq 18 \rangle \rangle
\end{aligned}
$$

In fact, the subject can have other attributes including specific ID number, associated risk level, etc., meanwhile, we skip these attributes for simplicity.

2.2 Cyber Threat Intelligence

Nowadays, many organizations produce, collect and share information related to cyber threats. According to National Institute of Standards and Technology (NIST)(Johnson et al., 2017), *Cyber Threat Information* is any information related to threats that might help organizations in protecting themselves against cyber threats or in detecting the activities of a threat agent. While, *Cyber Threat Intelligence* (hereinafter CTI), is what threat information becomes after its processing and analysis. Another definition given by Gartner considers CTI as *evidence-based knowledge that includes context, mechanisms, indicators, implications and actionable advice about an existing or emerging menace or hazard to IT or information assets*. Many organizations (e.g., NIST[1], MITRE[2]) have formulated enumerations of types of malware, vulnerabilities, and exploitations. Particularly, MITRE maintains three dictionaries, namely Common Vulnerabilities and Exposures (CVE)[3], Common Platform Enumeration (CPE)[4] and Common Weakness Enumeration (CWE)[5]. Additionally, the classification of attack patterns provided within Common Attack Pattern Enumeration and Classification (CAPEC)[6], and standardized in OASIS, XML/JSON-structured language for CTI representation i.e. *Structure Threat Information Expression* (STIX)[7].

Nowadays, the STIX standard is the most widely in use language for describing CTI in a structured way. The CTI report described through the STIX 2.0 standard is a collection of multiple *STIX Domain Objects* (SDOs) and *STIX Relationship Objects* (SROs)[8] that describe different concepts in domain and how those concepts relate to each other. Each SDO is

[1]https://www.nist.gov/
[2]https://www.mitre.org
[3]https://cve.mitre.org/
[4]https://cpe.mitre.org/
[5]https://cwe.mitre.org/
[6]https://capec.mitre.org/
[7]https://oasis-open.github.io/cti-documentation/
[8]http://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part2-stix-objects.html

based on a set of attributes known as *properties* (Intelligence, 2020). Each property defines specific information related to cyberattack including vulnerability ID, malware name, a pattern for a file with a given hash, textual recommendation for a countermeasure, etc. Furthermore, some properties can contain information related to a specific entity (e.g., adversary or victim) that allows security specialist using those properties for further mitigation strategies. The final report with the description of a cyberattack and described throught the STIX standard is represented as a collection of SDOs and SROs.

3 SECURITY POLICIES AND SECURITY REPORTS

In this section, we describe relationships between access control policy components and CTI elements using ontology for cybersecurity. As a backbone taxonomy, in this work, we adapt the *DOLCE-spray* (Oltramari et al., 2013) ontology that is a simplified version of *Descriptive Ontology for Linguistic and Cognitive Engineering* (DOLCE) (Masolo et al., 2002).

DOLCE aims at capturing the conceptual primitives underlying natural language and commonsense reasoning. DOLCE-spray hierarchy defines *entity* as at the class of anything that can be identifiable as an object. The *entity* is a root component that is defined in *concrete entity* and *abstract entity*. The *concrete entity* instances are located in definite spatiotemporal regions, while instances of *abstract entity* do not have inherent spatiotemporal dimensions. The *concrete entity* defines *continuant*, *occurrent*, and *physical quality* that are entities with inherent spatial parts, entities with inherent temporal parts and entities whose existence depends on their host respectively. The *continuant* further defines *agent*, *object* and *substance*. The *agent* describes *person* and *group*, while *object* defines *artifact* and *natural entity*.

The *abstract entity* branch of DOLCE-spray's root defines *abstract quality*, *information* and *characterization*. Furthermore, the *characterization* defines *role*, *plan*, *task* and *requirement*. The *abstract quality* class defines qualities that do not have any spatiotemporal dimension, while *information* refers to any content that can be conveyed by some physical *object*. The *characterization* defines a class for functions that map n-tuples of individuals to truth-values. Antisymmetric relation *characterizes* associates them with the objects they denote. Finally, the *requirement* class defines a set of demands either adopted or proposed by a *group*.

Following, we describe the alignment of the STIX objects, which represent CTI elements, and the DOLCE-spray ontology by using relations defined in the DOLCE-spray and using *Description Logic* (DL) notation. It is worth noting, that STIX does not have a specific *STIX Domain Object* to describe the system or network, instead, it conveys information about cybersecurity related entities such as files, systems, and networks using the *STIX Cyber-observables*[9]. Hence, for further definition of an asset i.e., a system that was attacked, we will use STIX Observed Data object denoted as $\texttt{Observed\_Data}_{sys}$. We used DOLCE-spray relations including *characterizes*, *isQualityOf*, *hasParticipant*, *uses*, *exploits* and *executes* to describe the alignment between DOLCE-spray and STIX standard.

$$\texttt{Identity}_a \sqsubseteq \texttt{ROLE} \sqcap \forall characterizes.\texttt{AGENT}$$
$$\texttt{Identity}_v \sqsubseteq \texttt{ROLE} \sqcap \forall characterizes.\texttt{AGENT}$$
$$\texttt{Identity}_v \sqsubseteq \neg\texttt{Identity}_a$$
$$\begin{aligned}\texttt{Observed\_Data}_{sys} \sqsubseteq\ & \texttt{ROLE} \\ & \sqcap \forall characterizes.\texttt{OBJECT} \\ & \sqcap \forall characterizes.\texttt{INFORMATION}\end{aligned}$$
$$\begin{aligned}\texttt{Malware} \sqsubseteq\ & \texttt{ROLE} \\ & \sqcap \forall characterizes.(\texttt{OBJECT} \sqcup \texttt{INFORMATION})\end{aligned}$$
$$\texttt{Malware} \sqsubseteq \neg\texttt{Observed\_Data}_{sys}$$
$$\texttt{Tool} \sqsubseteq \texttt{ROLE} \sqcap \forall characterizes.\texttt{OBJECT}$$
$$\texttt{Tool} \sqsubseteq \neg\texttt{Malware}$$
$$\begin{aligned}\texttt{Vulnerability} \sqsubseteq\ & \texttt{ABSTRACT QUALITY} \\ & \sqcap isQualityOf.\texttt{Observed\_Data}_{sys}\end{aligned}$$
$$\begin{aligned}\texttt{Attack\_Pattern} \equiv\ & \texttt{PLAN} \\ & \sqcap \exists hasParticipant.\texttt{Malware} \\ & \sqcap \exists hasParticipant.\texttt{Tool} \\ & \sqcap \exists exploits.\texttt{Vulnerability}\end{aligned}$$
$$\begin{aligned}\texttt{Threat\_Agent} \equiv\ & \texttt{Identity}_a \\ & \sqcap \forall exploits.\texttt{Vulnerability} \\ & \sqcap \exists executes.\texttt{Attack\_Pattern} \\ & \sqcap \exists uses.(\texttt{Tool} \sqcup \texttt{Malware})\end{aligned}$$
$$\begin{aligned}\texttt{Indicator} \sqsubseteq\ & \texttt{ARTIFACT} \\ & \sqcap \forall characterizes.\texttt{Threat\_Agent}\end{aligned}$$
$$\texttt{Indicator} \sqsubseteq \texttt{ARTIFACT} \sqcap \forall characterizes.\texttt{Malware}$$
$$\texttt{Indicator} \sqsubseteq \texttt{ARTIFACT} \sqcap \forall characterizes.\texttt{Identity}_a$$
$$\begin{aligned}\texttt{Course\_of\_Action} \sqsubseteq\ & \texttt{ARTIFACT} \\ & \sqcap hasParticipant.\texttt{PLAN} \\ & \sqcap hasParticipant.\texttt{INFORMATION} \\ & \sqcap mitigates.\texttt{Attack\_Pattern} \\ & \sqcap mitigates.\texttt{Vulnerability} \\ & \sqcap mitigates.\texttt{Malware}\end{aligned}$$

[9]https://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part4-cyber-observable-objects.pdf

Since the STIX `Identity` object allows defining entities that suffer from cyberattacks as well as entities that act in malicious attempts i.e., adversaries, we used $\texttt{Identity}_a$ and $\texttt{Identity}_v$ to differentiate adversary and victim entities. Hence, an adversary $\texttt{Identity}_a$ is described as an agent that is characterized by its *role*, while $\texttt{Identity}_v$ is an entity that suffers from the cyberattack and tries to protect assets. Differently to the `Identity` object, the `Threat_Agent` describes the profile of an adversary, that may belong to multiple entities. In this case, entities may use the same attack technique, strive to achieve the same goal (e.g., financial, personal gain, etc.), have same available resources level, etc. Thus, we define the `Threat_Agent` as $\texttt{Identity}_a$ that *exploits* `Vulnerability`, *executes* an `Attack_Pattern` and *uses* `Tool` in order to deliver `Malware`. In fact, both `Malware` and `Tool` are software variants, however, designed for different purposes. We define the `Attack_Pattern` as a `Plan` that defines `Malware` and `Tool` required to *exploit* the `Vulnerability`, while the `Vulnerability` is an *abstract quality* of a system denoted as $\texttt{Observed\_Data}_{sys}$. Finally, the countermeasure used to protect the system against cyberattacks or minimize potential impact is defined as the `Artifact` that provides the mitigation `Plan` together with necessary `Information` in order to *mitigate* `Attack_Pattern`, existing `Vulnerability` and `Malware` used to exploit it. In our work, we consider that any user can cause potential intentional or unintentional *threat* to a system. One of the security strategies to prevent a system from potential abuse and protect assets, is to define fine-grained access control policies and keep them up to date. Furthermore, some complex attacks similar to ones that used *Black-Energy* backdoor, were using multiple techniques, including the exploitation of privileged accounts. Thus, one of the mitigations of such attacks is to implement and use strong *Identity and Access Management* (IAM) controls together with fine-grained access policies to minimize or eliminate loses caused by illegal exploitation of privileged accounts. Therefore, following, we describe the alignment of basic components of ABAC model and DOLCE-spray ontology.

$$\texttt{USER} \sqsubseteq \texttt{ROLE} \sqcap \forall \textit{characterizes}.\texttt{AGENT}$$
$$\texttt{DEVICE} \sqsubseteq \texttt{ROLE}$$
$$\sqcap \forall \textit{characterizes}.(\texttt{ARTIFACT} \sqcup \texttt{INFORMATION})$$
$$\texttt{SUBJECT} \sqsubseteq \texttt{ROLE} \sqcap \forall \textit{hasParticipant}.(\texttt{USER} \sqcup \texttt{DEVICE})$$
$$\texttt{OBJECT} \sqsubseteq \texttt{ROLE}$$
$$\sqcap \forall \textit{characterizes}.(\texttt{ARTIFACT} \sqcup \texttt{INFORMATION})$$
$$\texttt{OBJECT} \sqsubseteq \neg \texttt{DEVICE}$$
$$\texttt{OPERATION} \sqsubseteq \texttt{ACTION} \sqcap \exists \textit{exploits}.\texttt{OBJECT}$$
$$\texttt{ABAC\_POLICY} \sqsubseteq \texttt{POLICY} \sqcap \exists \textit{protects}.\texttt{OBJECT}$$

In some scenarios, additionally to a user ID, a role and other attributes, an IP address or network type might be required to evaluate the user's request. Thus, in our work, we consider a user and device as independent elements of the ABAC model, since different users can request certain operations on an object using the same device as well as one user can perform requests using different devices. Hence, a subject is defined through the set of attributes that belong to a user and device. Elements of the ABAC policy model and STIX objects that constitute CTI reports are defined through their attributes. Since *Indicators of Compromise* (IoC) (e.g., IP and email addresses, names, etc.) specified in CTI reports can identify adversaries, these *identifiers* can be also used in ABAC policies to restrict access to users under certain conditions specified through IoC. Moreover, CTI reports also provide other information including time-window, when the malicious event occurred, and the number of observations of this event. This information can be used in defining the access control context of the policy. Furthermore, many CTI reports provide textual recommendations known as countermeasures that can be used either for eliminating or mitigating cyberattacks. However, countermeasures specified in the CTI reports provide a guidelines or a sequence of actions to be taken for protecting resources, rather than modifying security policies. Therefore, in our work, we use IoC to extend policies with new conditions, and we propose a method for extracting and enforcing mitigation strategies specified in CTI reports.

4 DEFINING ACCESS CONTROL POLICY UPDATE

In this section, we describe the access control policy update as a process that changes the state of the policy. In fact, any changes in access control policies can increase or decrease access rights of a certain subject, hence allowing previously restricted operations or restricting operations that were allowed by the policy. Furthermore, in some scenarios, a policy update process might require the creation of a new element to restrict some operations to be performed on an object if certain conditions are not satisfied. Hence, the process we consider includes 3 possible operations on the policy i.e., *policy update*, *policy creation* and *policy deletion*. Following we describe each operation of the process.

The *policy update* operation in further divided in *policy attenuation* and *policy restriction*. We consider the *policy attenuation* as an operation that increases privileges of a subject, while the *policy re-*

striction decreases subject's privileges by changing or removing one or multiple policy elements. Let the policy object $P_i := \langle SA_i, OA_i, EA_i, OP_i \rangle$ to be changes to $P_i^{'} := \langle SA_i^{'}, OA_i, EA_i^{'}, OP_i \rangle$. Hence, the policy object P_i is changed to $P_i^{'}$ by transforming the SA_i and EA_i to $SA_i^{'}$ and $EA_i^{'}$ respectively. Particularly, we consider the policy update operation changes access control conditions specified in the policy object without changing allowed operations specified in OP_i. Supposing that the policy 1 is changed to the policy 2, the privilege of the subject with a name Alice decreased since the level of associated risk defined in the policy decreased together with a time-window for accessing the *file1* and *file2* objects. Hence, operations *read* and *write* are no longer allowed for the subject Alice except the defined time-window from 11 to 15, and if the associated risk level is less or equal to 5. Thus, this is an example of *policy restriction* operation, since access privileges of the subject has decreased. On the other hand, the reverse operation, will increase the access privileges, thus it can be treated as the *policy attenuation* operation.

$$\begin{aligned}P_i :=& \langle \langle \exists name.Alice \wedge riskLevel \leq 8 \rangle \wedge \\ & \langle \exists object.file1 \wedge object.file2 \rangle \wedge \\ & \langle \exists currentTime \geq 9 \wedge currentTime \leq 18 \rangle \wedge \\ & \langle \exists operation.read \wedge operation.write \rangle \rangle\end{aligned} \tag{1}$$

$$\begin{aligned}P_i^{'} :=& \langle \langle \exists name.Alice \wedge riskLevel \leq 5 \rangle \wedge \\ & \langle \exists object.file1 \wedge object.file2 \rangle \wedge \\ & \langle \exists currentTime \geq 11 \wedge currentTime \leq 15 \rangle \wedge \\ & \langle \exists operation.read \wedge operation.write \rangle \rangle\end{aligned} \tag{2}$$

While the *policy attenuation* and *policy restriction* operations target modifications of certain policy elements, the *policy creation* targets creating the new policy according to available policy templates. The *policy creation* operation uses available policy templates to create the $P_i \equiv \langle SA_i, OA_i, EA_i, OP_i \rangle$ policy. Considering only positive authorization policies with *default* deny effect, the *policy creation* can be considered as a *policy attenuation* operation. Thus, before creating the policy P_i, the subject S_i could not perform any operation on the object O_i. Hence, we denote P_f as the *fictitious policy* that denies all operation requested by a subject S_i to be enforced on the object O_i. Thus, the *policy creation* can be treated as the *policy attenuation* operation, when the policy P_f is changed to the P_i policy.

Finally, the *policy deletion* operation is used for removing security policies from the database to eliminate the policy conflict caused by different effects of those policies for a single user. However, the *policy deletion* operation can be treated as the *policy restriction* operation since by removing the policy $P_i \equiv \langle SA_i, OA_i, EA_i, OP_i \rangle$ might result in restriction of some operations for the subject S_i.

5 PROPOSED FRAMEWORK

In this section, we present and describe our framework shown in Figure 1, the architecture of the tool for security management, and its deployment. The proposed framework consists of three main components following described.

The first component of the framework is the *Amazon Web Services* (AWS) account that provides a variety of cloud services to which access is controlled according to implemented RBAC and ABAC (Zhang et al., 2015) models. Particularly, the management of the AWS account is done with the *Identity and Access Management* (IAM[10]) service, that allows customers creating or removing users, assigning them individual credentials, or request temporary security credentials to provide users access to other AWS services and resources.

The second component of the framework is *Threat Intelligence Sharing Platform* (TISP) that acts as the black box allowing users to store and share CTI in the form of structured reports under security constraints specified in *Data-Sharing Agreements* (DSA) and defined by the *data controller*. Hence, only authorized entities can use shared and sanitized form of CTI reports. To define and enforce DSA, in (Martinelli et al., 2018) and (Osliak et al., 2019), authors proposed an approach used for the DSA representation and enforcement of *Data Operation Manipulation* specified in that DSA.

The third component of the framework is the *Security Management Tool* (SMT) that retrieves CTI reports from the TISP, and it also communicates with the AWS account. Following, we describe the SMT, its elements, and functionalities.

5.1 Security Management Tool

This sub-section reports and describe a component of the proposed framework. The core component of the framework is the *Security Management Tool* (SMT) that includes 9 elements as shown on Figure 2. The *CTI Manager* element is in charge of performing multiple retrieval operations on CTI reports specified according to the STIX standard. The element interacts with the *Context Analyzer* and *Attribute Engine*

[10]https://aws.amazon.com/iam/

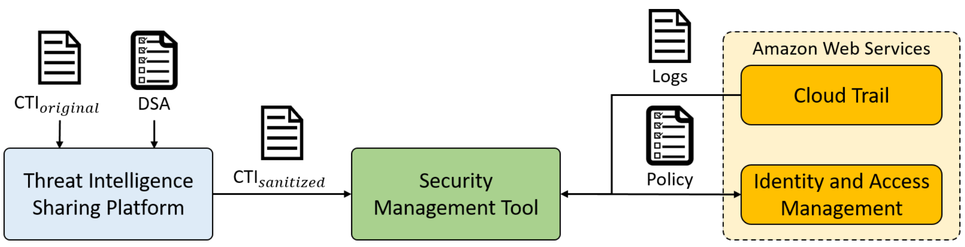

Figure 1: The proposed framework.

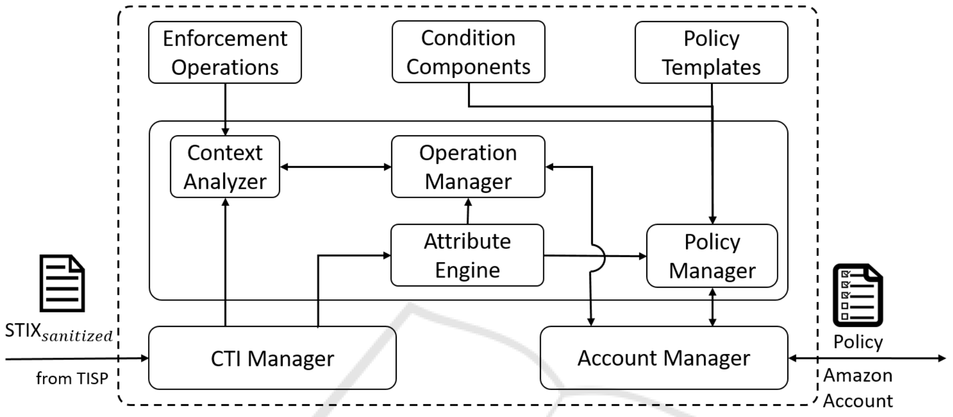

Figure 2: Security Management Tool.

elements by respectively forwarding *Course of Actions* (CoA) and attributes retrieved from the STIX report. The *Context Analyzer* is in charge of performing analysis of the CoA specified in the text format and selecting the most appropriate functions to be performed within the AWS account. The element exploits the advanced *Natural Language Processing* (NLP) approach offered by the *Spacy*[11] library. Hence, the *Context Analyzer* uses a combination of multiple functions used for the text analysis, including *Part-of-Speech* (PoS) tagging, *Dependency Parsing*, and *Semantic Similarity*. Particularly, the *Context Analyzer* element extracts root verb and noun pairs and matches each pair with the list of function names (e.g., remove user, attach policy, etc.) stored in the *Enforcement Operations* element, using semantic similarity determined by the word vectors comparison, selecting only those function names, whose similarity is in the range from 98% to 100% in order to not select random functions. Additionally, the *Context Analyzer* can use the *object of preposition* (e.g., table, group, etc.) and related *adjective* to improve the result of semantic similarity. Once the *Context Analyzer* defines the most appropriate functions, it forwards them to the *Operation Manager* element that is in charge of enforcing operations in the AWS account. As the function arguments, the *Operation Manager* element uses attributes from the *Attribute Engine* and it can also use additional information specified in the CoA. The *Attribute Engine* element stores retrieved attributes including IoC, names, and the number of occurrences of a single cyber event within a certain time-window. These attributes used by the *Operation Manager* and *Policy Manager* elements in enforcing functions within the AWS account and for policy operations respectively. Hence, the *Policy Manager* is in charge of updating existing and creating new policies according to available templates and elements of policy conditions stored in the *Policy Template* and *Condition Components* respectively. Finally, the *Account Manager* element acts as an interface of the AWS account and it is able to retrieve, create and upload policies, enforce operations specified in the *Operation Manager* element, and retrieve account activity using AWS *CloudWatch*[12] service. The CloudWatch service allows monitoring activity within the AWS account and use log files for different security purposes.

5.2 Practical Example

To better explain and demonstrate the functionality of our approach, in this section we illustrate an example of countermeasure specified in the CTI report.

To exploit vulnerabilities and steal information,

[11]https://spacy.io

[12]https://docs.aws.amazon.com/cloudwatch/

adversaries may manipulate accounts to maintain access to victim systems while acting as legitimate users (CERT, 2018) and perform iterative password updates to bypass password duration policies to preserve the life of compromised credentials (Ismail et al., 2015). However, adversaries must already have sufficient permissions on systems in order to create or manipulate accounts. Thus, potentially, adversaries may use the compromised accounts of system administrators. One of the solutions for this attack is to limit the privileges of those accounts either by updating security policies or removing certain user account from the system.

Therefore, in our work, in order to extend access control policies within the AWS account, we used malicious IP addresses shared with the MISP[13] platform, while in order to extract a sequence of security operations and enforce them within the AWS account, we used the Mitre Att&ck (Strom et al., 2018) framework that provides real-world observations of adversary tactics and techniques as well as related countermeasures. Particularly, we used *synthetic* CTI reports that include IoC, a description of the technique used in attack implementation and countermeasures to prevent this attack. Hence, the CTI report we considered denotes that the adversary uses a device with certain IP address in order to access the system and exploits *Account Manipulation*[14] technique to obtain confidential information, while one of the countermeasures[15] is to remove a user from a system. Figure 3 illustrates the *Privileged Account Management* countermeasure that addresses 80+ techniques used for cyberattacks including the aforementioned scenario. In this example, we retrieved the *remove users*

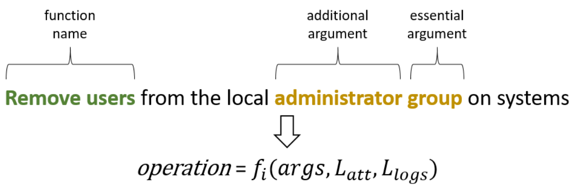
function name
additional argument
essential argument
Remove users from the local administrator group on systems
operation = $f_i(args, L_{att}, L_{logs})$

Figure 3: Countermeasure: example.

function name and compare it with the available functions in AWS by using word vector similarity to select a sequence of functions that have maximum similarity with retrieved function name. As the essential and additional function arguments, we use *group* and *administrator* respectively in order to define a domain where the function has to be enforced, and we compared these functions arguments with group names

[13]https://www.misp-project.org

[14]https://attack.mitre.org/techniques/T1098/

[15]https://attack.mitre.org/mitigations/M1026/

available in the AWS account. Moreover, the selected function also use a set of attributes retrieved from CTI reports. In fact, with our approach, we were able to retrieve more function names in other countermeasure strategies. However, due to the specificity of operations available in AWS IAM, with our approach, we were able to enforce limited number of operations retrieved from the CTI reports.

Finally, in our work, we did not consider platform/service-related CTI reports that provide more specific information about potential cyberattacks and countermeasures to be taken to prevent or mitigate those attacks. On the other hand, organizations can use specific properties of STIX objects (e.g., labels, custom properties) to specify the platform/service, to which the countermeasure has to be enforced. Meanwhile, by using the proposed approach, security specialist can create and enforce custom operations in other domains rather than AWS account management.

6 RELATED WORK

Although many works appear in the area of security policies, the policy updates process received little attention. In (Ray and Xin, 2004) authors proposed an algorithm for policy updates process that includes 4 possible operations on a policy object. Meanwhile, the authors considered simple authorization policies without taking into account any contextual information like date, time, etc. Furthermore, the proposed policy update process modifies only access rights i.e., actions allowed for a subject on a single object. Moreover, the usage of external knowledge for updating policies was out of the scope. While in our work, we consider more complex policies that define subjects, objects, actions, and access context through the set of corresponding attributes. Hence, in our work, the policy update process changes multiple policy elements rather than actions only. Furthermore, the policy update is implemented according to external knowledge in potential cyberattacks reported in a form of structured reports.

In another work (Yang et al., 2014), the authors proposed multiple policy updating algorithms for different types of access policies within the Cloud infrastructure. Meanwhile, this work does not use any information useful for policies update, thus restricting access to users that potentially threatening the system. However, in our work, we also describe a correlation between security policies and security reports elements, and we update policies according to collaborative CTI.

7 CONCLUSION

Updating security policies is a crucial task for any organization. In this study, we present and describe a framework that aims at supporting security policymakers within the Cloud infrastructure by providing possible actions to be taken for improving current IAM using collaborative knowledge in cybersecurity. The proposed framework uses collaborative and anonymized CTI shared through the threat intelligence sharing platform and preprocesses CTI with the proposed tool that exploits NLP approach. We described a correlation between elements of attribute-based access control policies and security reports using the most comprehensive ontology in cybersecurity. We present and describe the policy update process that modifies policy elements using IoC reported in security reports.

In future work, we plan to extend our approach to handle more complex attribute-based access control policies used in specific environments e.g., industrial sector, and enforce countermeasures specified in CTI. Moreover, we consider extending our framework with additional components for evaluating the effect of the updated policies on the overall system's security, and thus, test the efficiency of new policies. Hence, the future version approach will allow authorized users to access digital resources while preserving any access from entities defined in CTI reports.

ACKNOWLEDGMENT

This paper was partially supported by the EU H2020 funded project SPARTA, ga n. 830892.

REFERENCES

CERT, U. (2018). Russian government cyber activity targeting energy and other critical infrastructure sectors. *Us Cert*, pages 1–19.

Intelligence, O. C. T. (2016 (accessed August 28, 2020)). *STIX 2.0 Specification*.

Ismail, Z., Leneutre, J., and Fourati, A. (2015). An attack execution model for industrial control systems security assessment. In *Security of Industrial Control Systems and Cyber Physical Systems*, pages 157–167. Springer.

Jin, X., Krishnan, R., and Sandhu, R. (2012). A unified attribute-based access control model covering dac, mac and rbac. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 41–55. Springer.

Johnson, C., Badger, M., Waltermire, D., Snyder, J., and Skorupka, C. (2016). Guide to cyber threat information sharing. Technical report, National Institute of Standards and Technology.

Johnson, C., Feldman, L., Witte, G., et al. (2017). Cyber-threat intelligence and information sharing. Technical report, National Institute of Standards and Technology.

Martinelli, F., Osliak, O., and Saracino, A. (2018). Towards general scheme for data sharing agreements empowering privacy-preserving data analysis of structured cti. In *Computer Security*, pages 192–212. Springer.

Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., and Schneider, L. (2002). The wonderweb library of foundational ontologies.

Oltramari, A., Vetere, G., Chiari, I., Jezek, E., Zanzotto, F. M., Nissim, M., and Gangemi, A. (2013). Senso comune: A collaborative knowledge resource for italian. In *The People's Web Meets NLP*, pages 45–67. Springer.

Osliak, O., Saracino, A., and Martinelli, F. (2019). A scheme for the sticky policy representation supporting secure cyber-threat intelligence analysis and sharing. *Information & Computer Security*.

Ray, I. and Xin, T. (2004). Implementing real-time update of access control policies. In *Research Directions in Data and Applications Security XVIII*, pages 65–80. Springer.

Sandhu, R. S. (1998). Role-based access control. In *Advances in computers*, volume 46, pages 237–286. Elsevier.

Strom, B. E., Applebaum, A., Miller, D. P., Nickels, K. C., Pennington, A. G., and Thomas, C. B. (2018). Mitre att&ck: Design and philosophy. *Technical report*.

Yang, K., Jia, X., Ren, K., Xie, R., and Huang, L. (2014). Enabling efficient access control with dynamic policy updating for big data in the cloud. In *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pages 2013–2021. IEEE.

Zhang, Y., Patwa, F., and Sandhu, R. (2015). Community-based secure information and resource sharing in aws public cloud. In *2015 IEEE Conference on Collaboration and Internet computing (CIC)*, pages 46–53. IEEE.