Received 27 September 2022, accepted 22 December 2022, date of publication 30 December 2022, date of current version 5 January 2023.

Digital Object Identifier 10.1109/ACCESS.2022.3233404

RESEARCH ARTICLE

Fronesis: Digital Forensics-Based Early Detection of Ongoing Cyber-Attacks

ATHANASIOS DIMITRIADIS[1,2], EFSTRATIOS LONTZETIDIS[3], BOONSERM KULVATUNYOU[2], NENAD IVEZIC[2], DIMITRIS GRITZALIS[4], AND IOANNIS MAVRIDIS[1]

[1]Department of Applied Informatics, University of Macedonia, 54636 Thessaloniki, Greece
[2]Engineering Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899, USA
[3]Encode Centre of Excellence, 15124 Athens, Greece
[4]Department of Informatics, Athens University of Economics and Business (AUEB), 10434 Athens, Greece

Corresponding author: Dimitris Gritzalis (dgrit@aueb.gr)

This work was supported in part by a grant offered by the Ministry of Digital Governance of Greece to the Research Center of the Athens University of Economics and Business, Greece (2022-24).

ABSTRACT Traditional attack detection approaches utilize predefined databases of known signatures about already-seen tools and malicious activities observed in past cyber-attacks to detect future attacks. More sophisticated approaches apply machine learning to detect abnormal behavior. Nevertheless, a growing number of successful attacks and the increasing ingenuity of attackers prove that these approaches are insufficient. This paper introduces an approach for digital forensics-based early detection of ongoing cyber-attacks called Fronesis. The approach combines ontological reasoning with the MITRE ATT&CK framework, the Cyber Kill Chain model, and the digital artifacts acquired continuously from the monitored computer system. Fronesis examines the collected digital artifacts by applying rule-based reasoning on the Fronesis cyber-attack detection ontology to identify traces of adversarial techniques. The identified techniques are correlated to tactics, which are then mapped to corresponding phases of the Cyber Kill Chain model, resulting in the detection of an ongoing cyber-attack. Finally, the proposed approach is demonstrated through an email phishing attack scenario.

INDEX TERMS Cyber-attack detection, cyber kill chain, cybersecurity, digital artifacts, MITRE ATT&CK, ontology, rule-based reasoning.

I. INTRODUCTION

Dealing with cyber-attacks is a critical factor for organizations to achieve their business goals; it is a business-driven factor rather than a best practice. To address cyber-attacks, one of the five cybersecurity functions is detection, as defined in the Cybersecurity Framework developed by the Information Technology Laboratory (ITL) of the National Institute of Standards and Technology (NIST) [1]. Detection can take place before a cyber-attack is launched (threat detection), where attackers perform reconnaissance and weaponization activities. It can also take place during a cyber-attack (early detection of ongoing cyber-attack) or after a cyber-attack is accomplished (post-compromise detection), where intruders accomplish their original objectives [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Yu.

Detection approaches are classified as statistics-based (i.e., anomaly or behavioral-based), pattern-based, rule-based, state-based, and heuristic-based [3]. Statistics-based approaches create the profile of a monitored system and use this profile to detect cyber-attacks as abnormal activities that are beyond an ordinal baseline. Pattern-based approaches look for specific predefined patterns of data or actions to detect cyber-attacks. Rule-based approaches run a set of rules against a system to conclude the presence of a cyber-attack. Rules are usually implemented as if-then statements to construct models of malicious behaviors. Compared to patterns, rules are more easily extended and maintained since they do not predefine strict huge patterns. State-based approaches utilize finite state machines to construct a cyber-attack detection algorithm. Finally, heuristic-based approaches use a model, a decision-making algorithm, and a set of conditions and rules.

Current detection approaches seem to face significant issues in the early detection of ongoing cyber-attacks, and more effort should be put into this area. According to Mandiant's threat report, only 59% of the security incidents were detected by the organizations themselves in 2020, while the median dwell time of an adversary within a compromised organization was 24 days [4]. In some cases, the dwell time can be even longer, especially when the attack is not detected by the organization itself but by external third parties. The insufficiency of the current detection approaches was also pointed out by MITRE [5].

To help organizations with building rule-based detection approaches, MITRE developed the Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) framework [5]. This framework defines the techniques that can be used by attackers to achieve short-term goals, called tactics, during a cyber-attack [5]. Therefore, a MITRE ATT&CK-based detection approach is expected to identify the techniques and the corresponding tactics operated within a system and to utilize them towards detecting a cyber-attack, as earliest as possible. Most approaches that utilize MITRE ATT&CK for detection purposes, as listed in [6], are limited to identifying the operation of techniques; namely, they do not use the identified techniques for cyber-attack detection. In addition, they do not utilize the wealth of forensics data, known as digital artifacts, produced during the system operation and their efforts are limited to examining data from event logs and network traffic captures [3].

Digital artifacts can provide much more information regarding user and system events since they can include both non-volatile (e.g., event logs, emails) and volatile data (e.g., processes, and RAM contents) [7]. Therefore, an efficient cyber-attack detection approach should consider enhanced utilization of digital artifacts. In addition, their acquisition from the monitored system should be performed with digital forensics practices to preserve their integrity. The proactive application of digital forensics practices before or during a cyber-attack has been already pointed out in the literature [8], [9], [10].

In this paper, a digital forensics approach for early detecting ongoing cyber-attacks, called Fronesis, is proposed. Fronesis combines ontological reasoning with the MITRE ATT&CK framework, the Cyber Kill Chain (CKC) model [2], and the digital artifacts acquired from the monitored computer system with digital forensics practices. The CKC model provides the sequence of phases of a cyber-attack, while the MITRE ATT&CK framework provides the techniques for accomplishing each phase. The operation of each technique leaves some traces (i.e., digital artifacts) in the monitored computer system. Fronesis examines the digital artifacts acquired from the monitored computer system to identify operating techniques. It then maps the identified techniques into the CKC phases that are used to reconstruct an ongoing cyber-attack based on the CKC model, resulting in the attack detection. The more phases of the CKC model are identified, the more accurate the detection. The proposed implementation of Fronesis includes rule-based reasoning on the Fronesis ontology. The Fronesis ontology represents the techniques and tactics of MITRE ATT&CK, the phases of CKC, and the attributes of digital forensics in a machine-understandable form. The proposed rule-based reasoning process allows expressing Fronesis detection logic declaratively and produces output in the form of instances of detected CKC phases of ongoing cyber-attacks. The applicability of the proposed approach is demonstrated through an email phishing attack scenario.

The contribution of this paper is a proposed detection approach for ongoing cyber-attacks, called Fronesis. The core of Fronesis is a multi-step methodology that was developed by integrating the CKC model and MITRE ATT&CK. The input of this multi-step methodology is digital artifacts acquired from a monitored system and the output is the reconstruction and the detection of a cyber-attack. The novelties of Fronesis are the following:

1) The mapping of the CKC model to MITRE ATT&CK in order to define the techniques that can be used for accomplishing each CKC phase. This overcomes the limitation of the CKC model regarding its lack in defining the techniques to operate each CKC phase.
2) The consideration of digital artifacts for recognizing the operation of a technique in a monitored system. Digital artifacts include volatile data, such as processes, and non-volatile data, such as emails, email attachments, log files and documents. As a consequent, they provide much more information than log files used by other detection approaches and so they can enable better detection results.
3) The reconstruction and detection of an ongoing cyber-attack using digital artifacts. This leads to digital forensics readiness which is the ability to collect evidence (i.e., digital artifacts) while minimizing the cost and time [11]. Since Fronesis reconstructs an ongoing cyber-attack using digital artifacts, the evidence is already collected and as a result, the time and the cost of their collection are minimized.
4) The detection of a cyber-attack rather than the detection of particular MITRE ATT&CK techniques. The current approaches related to Fronesis are limited to identifying particular MITRE ATT&CK only. Besides the automatic detection of MITRE ATT&CK techniques, Fronesis correlates them on the basis of digital artifacts in order to detect and reconstruct a cyber-attack in progress.
5) The notion of "Combinations Of Sequences of CKC Phases (COSPs)". A COSP describes a cyber-attack that is operated based on the CKC model but skips one or more CKC phases. Therefore, a COSP is necessary to describe and detect cyber-attacks that they do not strictly follow the CKC model. Fronesis defines three conditions that should be met in order to describe a cyber-attack using a COSP. These conditions are: (a) the order of CKC phases of a COSP should follow

the CKC model, and (b) the CKC phases of a COSP should be related and (c) subsequent.

Fronesis detects any cyber-attack that follows a combination of the CKC phases (i.e., COSP) and utilizes adversarial techniques defined in the MITRE ATT&CK. Therefore, the limitation of Fronesis is that it cannot detect cyber-attacks that their operation cannot be described by the CKC model or use a new adversarial technique that is not defined in MITRE ATT&CK yet.

The rest of the paper is arranged as follows. Section 2 provides the background necessary for describing the proposed approach. Section 3 presents the Fronesis approach. Section 4 describes the ontology and the rule-based reasoning that implement Fronesis. Section 5 demonstrates the application of Fronesis. Section 6 outlines the related work, and finally, Section 7 concludes with present and future efforts.

II. BACKGROUND

This Section presents the concepts necessary for presenting the subsequent Sections and the proposed approach.

A. CYBER KILL CHAIN

Lockheed Martin proposed the Cyber Kill Chain (CKC), which is an intelligence-driven model for protection and detection purposes. CKC defines the sequence of the phases that adversaries can follow to operate an attack and achieve their malicious objectives [2]. The CKC phases include the following:

1) Reconnaissance (R) which involves all actions for identifying and selecting the target. Some examples are information gathering through public information on websites, and port scanning.
2) Weaponization (W) which involves the development of a legitimate-looking file (e.g., docx, xls, doc) infected with malware. This phase occurs in the attacker's infrastructure and cannot be discerned by any defensive security tool or team [12].
3) Delivery (D) which includes the methods that can be used for delivering the above-mentioned legitimate-looking file to the target. Some examples of delivery methods are emails and USB devices.
4) Exploitation (E). In this phase, the payload (i.e., malware) of the legitimate-looking file exploits a vulnerability and runs in the target system.
5) Installation (I) where the payload installs itself or a malicious code in the compromised system in order to achieve permanent existence (i.e., it can run in the system automatically at a specific time or after an event such as a reboot).
6) Command and Control (C2) in which the payload communicates with the attacker. This communication usually takes place via a covert communication channel. For instance, the payload sends and receives information and commands via DNS queries.
7) Actions on Objective (A) where attackers accomplish their objectives (e.g., data exfiltration, lateral movement to other systems).

B. MITRE ATT&CK

MITRE ATT&CK is a publicly available knowledge base of the techniques utilized by adversaries to achieve their objectives [5], [13]. MITRE ATT&CK can be utilized for both offensive and defensive purposes, such as penetration testing and cyber-attack detection. MITRE ATT&CK consists of tactics, techniques, and procedures.

Tactics are short-term goals achieved during an attack. One example is the Initial Access tactic that is the short-term goal of gaining an initial foothold within a network or system. For each tactic, MITRE provides a detailed description of the tactic's goal.

Techniques are actions that adversaries perform to achieve a specific tactic. For instance, the "Phishing" technique can be used for achieving the "Initial Access" tactic by delivering malware to the target. For each technique, MITRE provides a detailed description of how it can be accomplished. Some techniques are broken down into more specific techniques to describe a more specific description of the parent technique. For instance, the "Phishing" technique is broken down into the "Spearphishing Link," "Spearphishing Attachment," and "Spearphishing via Service." Some techniques belong to more than one tactic because they can be used for achieving different goals. For instance, the technique "Scheduled Task/Job" can be used by malware to install itself to a system as a scheduled task but also to run automatically at a specific time. This technique, therefore, belongs to the "Persistence" and "Execution" tactics, respectively.

Procedures are implementations of techniques, namely how a technique can be actually operated. For instance, one procedure of the "Spearphishing Attachment" is an adversary sending an email containing a malicious Microsoft Office.doc attachment. For each technique, MITRE provides examples of procedures that describe in specific detail how the technique was carried out by a real-world reported case.

C. DIGITAL FORENSICS

Digital Forensics (DF) is the application of computer science to support the investigation, review, or prosecution of incidents that involve digital data. Such incidents can range from policy violation incidents to cyber-crimes and felonies. The outcome of DF is usually a report of evidentiary data of the incident [7]. The evidentiary data is used to answer the questions of who was involved in the incident, what happened, where the incident took place, when, and how it happened [7]. In other words, the evidentiary data are the traces of an incident, such as a cyber-attack. The evidentiary data is also called evidentiary artifacts or evidentiary digital artifacts. Although there is no formal definition of the term digital artifact [14], in this paper, a digital artifact is any digital data created or modified by an action of a human, software, or device. Some examples are emails, email attachments, registry keys, word documents, and IP addresses. Digital artifacts also include volatile data such as processes. The digital artifacts restored from a system constitute robust evidence about an incident under investigation, providing that their integrity is preserved during their acquisition [7].

III. THE PROPOSED APPROACH

The proposed detection approach, called Fronesis, utilizes the MITRE ATT&CK and CKC model to reconstruct an ongoing cyber-attack from digital artifacts [15]. More specifically, Fronesis starts by examining digital artifacts acquired from the monitored system in order to recognize the operation of MITRE ATT&CK techniques (hereinafter called just techniques). The digital artifacts are acquired with sensors that follow digital forensics practices to preserve their integrity. The recognized techniques are then utilized to identify MITRE ATT&CK tactics (hereinafter called just tactics). Afterward, Fronesis identifies the operation of CKC phases (hereinafter called just phases) based on the identified tactics and finally detects an ongoing cyber-attack by reconstructing it based on the CKC model.

A particular cyber-attack might consist of various combinations of phases since some phases of the CKC model may not be used. For example, there might be a case of delivering malware where there is no need to install it in the targeted system; so, the Installation phase is skipped. One example of such malware is UIWIX [16]. In this vein, Fronesis reconstructs an ongoing cyber-attack by detecting one of the following Combinations Of Sequences of CKC Phases (COSPs):

1) Delivery, and Exploitation – COSP(DE): An attacker gains access within a system via delivering malware that exploits a vulnerability.
2) Delivery, Exploitation, and Installation – COSP(DEI): In addition to DE, the malware obtains persistence in the compromised system.
3) Delivery, Exploitation, and C2 – COSP(DEC): In addition to DE, the malware communicates with the attacker for command-and-control purposes.
4) Delivery, Exploitation, Installation, and C2 – COSP(DEIC): In addition to DEC, the malware installs itself in the compromised system

Fronesis does not consider Reconnaissance and Weaponization since they are preparation phases [17]. According to MITRE, these phases are actions that attackers take before executing an attack or before trying to access a network [5]. MITRE considers the Reconnaissance and Weaponization (MITRE calls it as ''Resource Development'') phases in the PRE-ATT&CK framework which is focused on recognizing pre-attack actions for prevention purposes [5]. The Actions on Objective phase is also excluded since the detection should occur before the attackers achieve their objective in this phase.

A. CONCEPTS

The main concepts of Fronesis are: COSP, phases, techniques, tactics, and digital artifacts. The UML diagram in Figure 1 presents the relationships among the forenamed Fronesis concepts. More specifically, a digital artifact can be associated with other digital artifacts with the *hasRelatedDigitalArtifact* relationship. For instance, a file (e.g., docx document) is associated with the process that opened it (e.g., Microsoft Word), or an email message is associated with its attachments. The *hasRelatedDigitalArtifact* relationship can be specialized to convey more precise semantics as well. For example, a specialized *hasAttachedFile* relationship can be created when specifying the relationship between an email message and its attached file.

A technique can be associated with one or many digital artifacts with the *hasTrace* relationship. Indeed, the operation of a technique can leave one or more traces in a system. These traces are realized as digital artifacts. To identify them, the description of the technique as provided in the MITRE ATT&CK knowledge base is examined. For instance, the description of the technique ''Spearphishing Attachment'' mentions that the technique is accomplished via an email message that contains an attached file. This means that the digital artifacts, which are the traces of this technique, include an email message and an attached file.

A tactic can be associated with one and only one technique with the *hasTechnique* relationship because the execution of a particular technique can achieve only one specific tactic at a time.

A phase can be associated with a tactic with the *mapsTo* relationship. Indeed, the Delivery, Exploitation, Installation, and C2 phases are mapped to the Initial Access, Execution, Persistence, Command, and Control tactics, respectively, since they serve the same purpose. Fronesis uses these tactics only since its detection logic is based on the CKC model with respect to COSPs. Consequently, a COSP as used in Fronesis can be associated with two to four phases with the *hasPhase* relationship.

It should be noted that even though a tactic can be achieved with more than one technique, Fronesis creates and relates a new tactic instance for every recognized technique. In this way, a self-contained thread from a COSP to Digital Artifacts can be established such that only relevant traces (i.e., digital artifacts) and phases can be analyzed and decided to whether they constitute a COSP.

B. METHODOLOGY

Fronesis utilizes the relationships among the concepts described in subsection 3.1 to detect ongoing cyber-attacks. To do so, Fronesis follows a proposed multi-step methodology, which is explained below and depicted in Figure 2:

- Step 1: Preparation steps: The following steps 1.1 to 1.3 are repeated until all digital artifacts are examined. This process is depicted with the ''for all digital artifacts'' loop in Figure 2. The examination of all digital

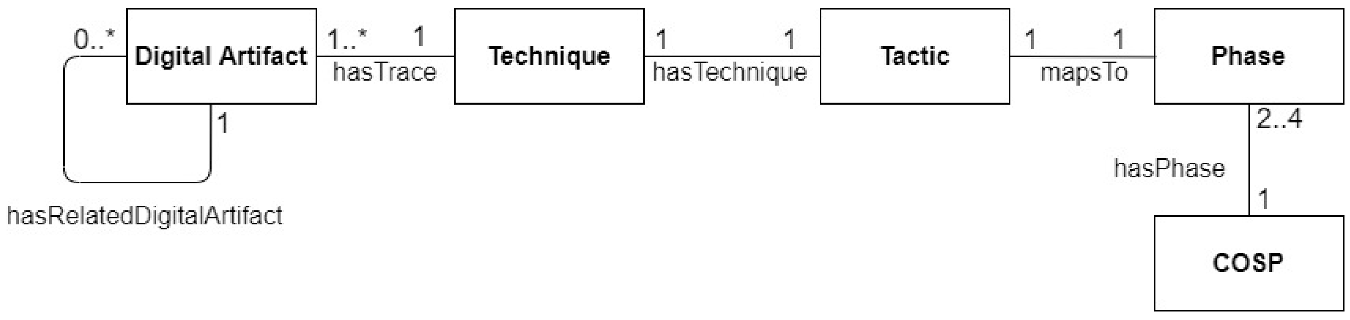

FIGURE 1. UML diagram of Fronesis concepts.

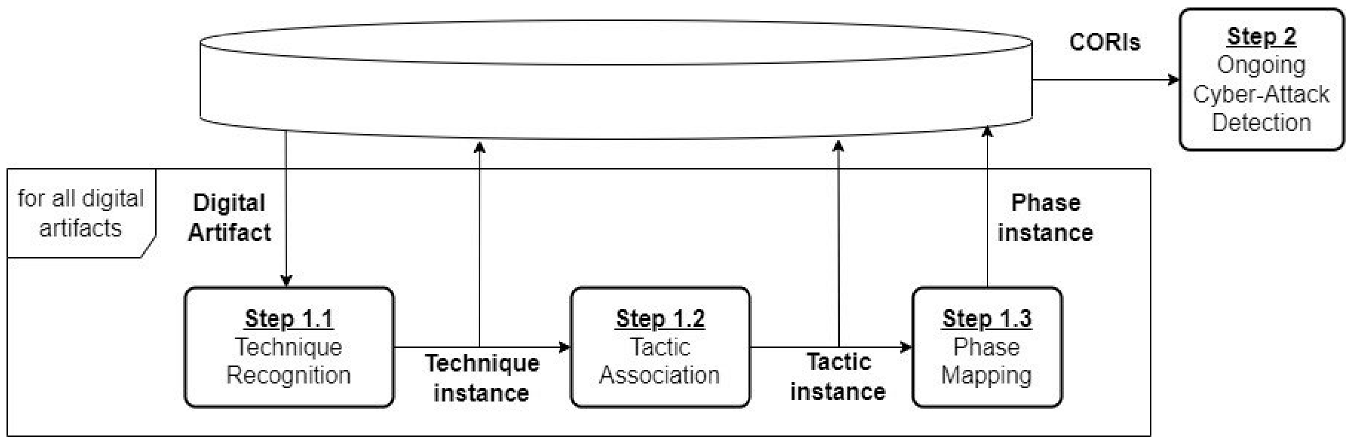

FIGURE 2. The proposed step-by-step methodology.

artifacts ensures the recognition of all the techniques operated within the monitored system. Note that the digital artifacts are acquired from the monitored system and provided beforehand. It is out of the scope of this study to recommend a specific application tool to acquire the digital artifacts from the monitored system.
o Step 1.1: Technique Recognition. The operation of a technique is recognized based on the digital artifacts that it creates in the monitored system during its operation. As a result, an instance of the technique is created. The *hasTrace* relationship is used to associate the new Technique instance with the digital artifacts used to recognize the technique.
o Step 1.2: Tactic Association. This step associates a new Tactic instance to the Technique instance from Step 1 using the *hasTechnique* relationship. Because a technique may be used in several tactics, multiple Tactic instances and relationships may be created.
o Step 1.3: Phase Mapping. The tactics identified in Step 2 are mapped to new Phase instances. The *mapsTo* relationship is used to associate each new Phase instance with a Tactic instance of Step 1.2.

- Step 2: Ongoing Cyber-Attack Detection. Steps 1.1 to 1.3 result in chains of recognized instances (CORI). Each CORI consists of a chain of one Phase instance, one Tactic instance, one Technique instance, and a set of Digital Artifact instances linked according to the discovery in Step 1. So, the traces of a Technique instance are the traces of a Phase instance. As detailed in the next subsection 3.3, in this Step 2, Fronesis utilizes CORIs to form a COSP. In essence, each formed COSP is the reconstruction and detection of an ongoing cyber-attack.

C. ONGOING CYBER-ATTACK DETECTION

Figure 3 depicts an analytical view of Step 2 of the methodology presented in subsection 3.2. Fronesis utilizes CORIs to form a COSP in order to reconstruct and detect an ongoing cyber-attack. A formed COSP should meet the following three conditions.

1) The combination and the order of Phase instances should match one of the COSPs described in subsection 3.1. For instance, a formed COSP(DE) should consist of a Delivery and an Exploitation Phase instances (the former precedes the latter in the CKC model).
2) All pairs of adjacent ***Phase instances should be related*** (e.g., both the DE and the EC pairs of a COSP(DEC)). **Def**. "Phase instances are related when their traces are correlated; and traces are correlated when they have attributes with common or temporally-related values." Examples of trace attributes are names,

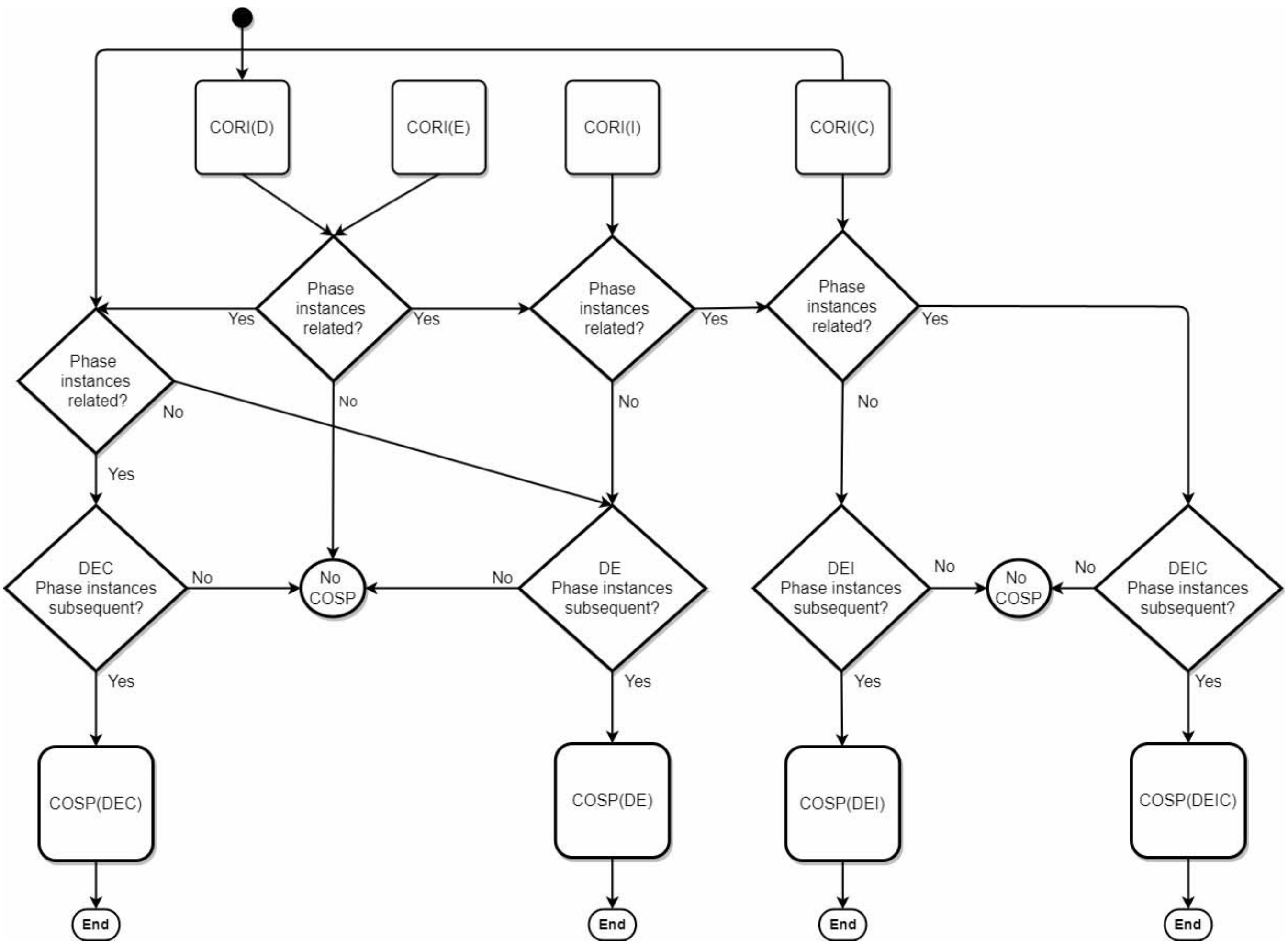
CORI(D)
CORI(E)
CORI(I)
CORI(C)
Phase instances related?
Yes
No
DEC Phase instances subsequent?
DE Phase instances subsequent?
DEI Phase instances subsequent?
DEIC Phase instances subsequent?
No COSP
COSP(DEC)
COSP(DE)
COSP(DEI)
COSP(DEIC)
End

FIGURE 3. **Step 2 of the methodology: Ongoing Cyber-Attack Detection.**

fullpaths and timestamps. An example of correlated phase instances identified by the fullpath attribute is when the downloaded attachment of an email message in a trace of a Delivery Phase instance has the same value as the fullpath attribute of an opened file of a process in a trace of an Exploitation Phase instance. Timestamp attributes are temporally-related when they are within a defined block of time (i.e., minor time difference).

3) All pairs of adjacent ***Phase instances must be subsequent***. **Def.** "Phase instances are subsequent when the correlated traces of a Phase instance are newer than the ones in its preceding Phase instance." Following the previous example, given that the attachment of the email message was downloaded at time $t1$ (timestamp attribute) and the process opened the file at time $t2$, the Delivery and Exploitation Phase instances are subsequent if $t2 - t1 \geq 0$.

As depicted in Figure 3, for each one CORI(D), Fronesis looks for a CORI(E) with a related Phase instance. If it is not found, Fronesis continues with the next CORI(D). If it is found, the related Phase instances are candidates to form a COSP(DE) or a COSP with another combination. Then, Fronesis looks for CORI(I) and CORE(C) with Phase instances related to the ones of the already located CORI(D) and CORI(E). The final form of the COSP is determined after Fronesis verifies that the Phase instances of all located CORIs are subsequent.

IV. IMPLEMENTATION

Fronesis can be implemented via different ways, such as a program code, machine learning and rule-based reasoning. In this paper, Fronesis was implemented via the initialization of a proposed ontology and rule-based reasoning. The Fronesis ontology describes the main concepts of Fronesis as well as their attributes and relationships. It was written in the Web Ontology Language (OWL) [18]. Rule-based reasoning implements the Fronesis methodology described in subsection 3.3 for ongoing cyber-attack detection. These declarative rules are written in the Semantic Web Rule Language (SWRL) [19]. Rule-based reasoning is achieved via the Drools rule engine [20]. Protégé [21] ontology development environment was used for the implementation and experiments.

A. FRONESIS ONTOLOGY

The domain of the proposed ontology is cyber-attack detection, with a description of the concepts of the COSPs, phases, techniques, tactics, and digital artifacts, as well as their attributes and relationships. The following subsections describe the classes in the proposed ontology as well as data and object properties.

1) CLASSES

The primary classes of the proposed ontology are described below. They are all disjoint classes.

- *Phase* class. It has *Delivery*, *Exploitation*, *Installation*, and *CommandAndControl* subclasses representing the four types of phases.
- *Tactic* class. It has *MA_InitialAccess*, *MA_Execution*, *MA_Persistence*, *MA_CommandAndControl* subclasses representing four types of tactics.
- *Technique* class. It has subclasses representing the various types of techniques related to the tactic subclasses above. For instance, the techniques of the *Privilege Escalation* tactic are not included.
- *Artifact* class. It has subclasses representing various types of digital artifacts such as *WindowsTask*, *File*, and *EmailMessage*. These subclasses are based on the Unified Cyber Ontology (UCO) 0.7.0 Release [22]. UCO is a community-developed ontology for the cyber security domain that includes representations of digital artifacts along with their data properties.
- *COSP* class. It represents the reconstruction of a detected ongoing cyber-attack based on a COSP. Each *COSP* individual (i.e., an instance of the *COSP* class) is associated with the CORIs that form the corresponding COSP.

2) DATA PROPERTIES

Data properties or attributes connect an individual of a class with a specific value (e.g., a string or an integer value). To do so, each data property has a domain and range. The domain restricts the classes allowed to have the data property, and the range restricts the datatypes (e.g., *xsd:string*) or enumerated values allowed for the data property.

Data properties for *Artifact* subclasses were defined. Their domains are a particular *Artifact* subclass, and their ranges depend on the datatype that they describe (e.g., *string* or *integer*). Most data properties in *Artifact* subclasses are based on UCO. For instance, the data property *modifiedTime* with the *File* artifact domain and the *xsd:datetime* range was derived from UCO. However, additional data properties were also defined when it was found missing from UCO. Some examples are the data properties *hasFullPath*, *hasFileName*, *hasExtension* with the domain and range that are respectively *File* and *xsd:string*. These new data properties were defined based on the attributes of the digital artifacts. Among the most important data properties added are related to timestamps. Some examples are *createdTime*, *modifiedTime*, *accessedTime*, and *deletedTime* that represent the time that an *Artifact* individual was created, modified, accessed, or deleted, respectively. Timestamps are essential for checking whether the *Phase* instances of a COSP are correlated and subsequent.

3) OBJECT PROPERTIES

Object properties allow for creating relationships between two individuals (instances of classes). One individual is the domain value of the object property and the other is the range value. An object property is also known as a predicate that consists of a relationship and the subject and object of the relationship. The following object properties represent the relationships between *COSP*, *Phase*, *Tactic*, *Technique*, and *Artifact* classes. These relationships are also inherited by their subclasses.

- *hasPhase* whose domain is COSP, and range is *Phase*. Object restrictions *min cardinality* 2 and *max cardinality* 4 were defined such that a *COSP* individual must be associated with at least two and not more than four *Phase* individuals.
- *mapsTo* whose domain is *Phase* and range is *Tactic*. The object property restriction *exact cardinality* 1 was defined such that a *Phase* individual must be associated with one and only one *Tactic* individual.
- *hasTechnique* whose domain is *Tactic* and range is *Technique*. The object property restriction *exact cardinality* 1 was defined such that a *Tactic* individual must be associated with one and only one *Technique* individual.
- *hasTrace* whose domain is *Technique* and range is *Artifact*. The object property restriction *min cardinality* 1 was defined such that a *Technique* individual must be associated with at least one *Artifact* individual.

Additionally, object properties that are subproperties of *hasRelatedDigitalArtifact* were also defined to represent specific relationships between the *Artifact* subclasses explained in subsection 3.1. The following steps were taken to define these properties for all techniques:

1) Selection of a technique from MITRE ATT&CK. The selected technique should belong to one of the Initial Access, Execution, Persistence, and Command and Control tactics. For instance, the technique "Spearphishing Attachment" of the Initial Access tactic is selected.
2) *Artifact* subclasses creation. The description of the selected technique is examined to define the traces that the technique leaves in a system. These traces are in the form of digital artifacts and so individuals of *Artifact* subclasses. For instance, the description of the technique "Spearphishing Attachment" mentions that the technique is accomplished via an email message that contains an attached file. This means that the digital artifacts related to this technique are an email message and an attached file. These two digital

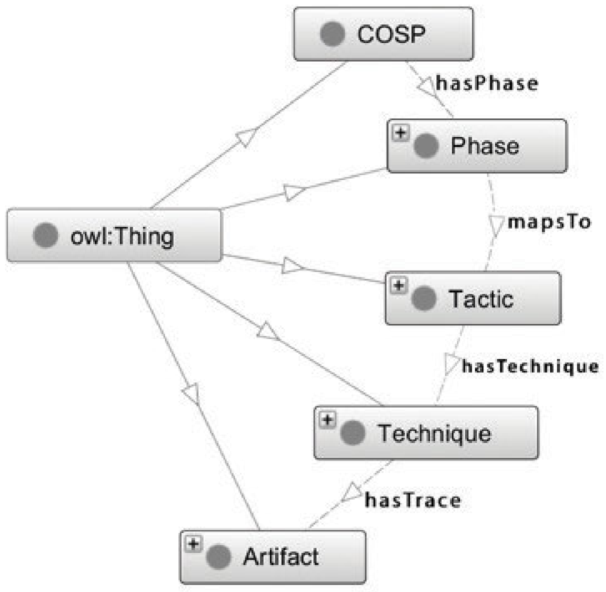

FIGURE 4. **OntoGraf rendering of the proposed ontology.**

artifacts are represented with the *EmailMessage* and *File* subclasses of the *Artifact* class.

3) Object properties creation. Based on the description of the selected technique, the relationships between the identified *Artifact* subclasses of the previous step are identified. These relationships are used to define object properties. Following the previous example, the email message may have an attached file. Therefore, the object property *hasAttachedFile* is defined. Its domain is the *EmailMessage* subclass, and its range is the *File* subclass.

Figure 4 depicts the five main classes in the proposed ontology and their relationships. For simplicity, their subclasses and their inherited relationships are omitted.

Consistency checking is also part of the ontology development process to validate the ontology. It ensures there are no logical conflicts or undesirable inferences across the classes, properties, and their domains and ranges. This checking was performed by the Pellet OWL reasoner plugin for Protégé [23].

4) DECLARATIVE RULES

Rules were developed to identify CORIs from artifact assertions and form COSPs that indicate ongoing cyber-attacks. Except for *Artifact* subclasses individuals, all other individuals, such as *Technique* subclasses individuals, are created by rules. *Artifact* subclass individuals along with their properties are asserted beforehand from digital artifacts acquired from the monitored system. Nevertheless, it is out of scope to recommend an application tool for that purpose.

The rules are an implementation of Fronesis methodology detailed in Section 3. Each rule consists of a condition (i.e., antecedent) and an action (i.e., consequent). When the condition is satisfied, the action is executed. The action part creates a new individual and performs property assertions. For instance, the condition part of rules that implement Step 1.2 of the Fronesis methodology would match a *Technique* subclass individual, and the action part would create a *Tactic* subclass individual and assert a *hasTechnique* relationship to the *Technique* subclass individual. The rules are categorized into the following four groups.

1) Rules for recognizing techniques. Each rule recognizes a particular technique from specific digital artifacts according to Step 1.1 described in subsection 3.2. These digital artifacts are individuals of *Artifact* subclasses, and the relationships among them are object properties. A technique is recognized when the specific *Artifact* subclass individuals and the relationships among them exist in the digital artifacts dataset provided beforehand. In this vein, the condition part checks for the presence of these *Artifact* subclass individuals. It also checks if there are the necessary relationships among them. When both of these conditions are met, the action part is activated to assert a new *Technique* subclass individual and the *hasTrace* object property to the *Artifact* subclass individuals that have activated the rule.
2) Rules for associating tactics. A rule was defined for each technique. The condition part checks for the presence of a *Technique* subclass individual; and the action part instantiates the proper *Tactic* subclass individual that is relevant to the *Technique* subclass individual by the *hasTechnique* object property.
3) Rules for mapping phases. A rule was defined for each type of phase. The condition part checks for the presence of an individual of a specific *Tactic* subclass based on the mapping between phases and tactics. The action part asserts an individual of the proper *Phase* subclass and asserts a *mapsTo* relationship to the *Tactic* individual.
4) Rules for detecting an ongoing cyber-attack. These rules follow Step 2 of the Fronesis methodology described in subsection 3.2. Multiple rules for each COSP were defined. The condition part checks if the three conditions of subsection 3.3. are met, and the action part asserts the corresponding *COSP* individual and the *hasPhase* relationship to the proper *Phase* subclass individuals. This process is explained in further detail below.

Each phase is accomplished via a specific technique considering that a phase maps to a tactic that the technique belongs to. Therefore, the Delivery, Exploitation, Installation, and C2 phases can be accomplished by about 19, 34, 100, and 38 techniques, respectively. Consequently, there are 646, 64,600, 24,548, and 2 million possible CORIs to form a COSP(DE), COSP(DEI), COSP(DEC), and COSP(DEIC), respectively. This means that the number of all possible scenarios and consequently the number of rules needed to

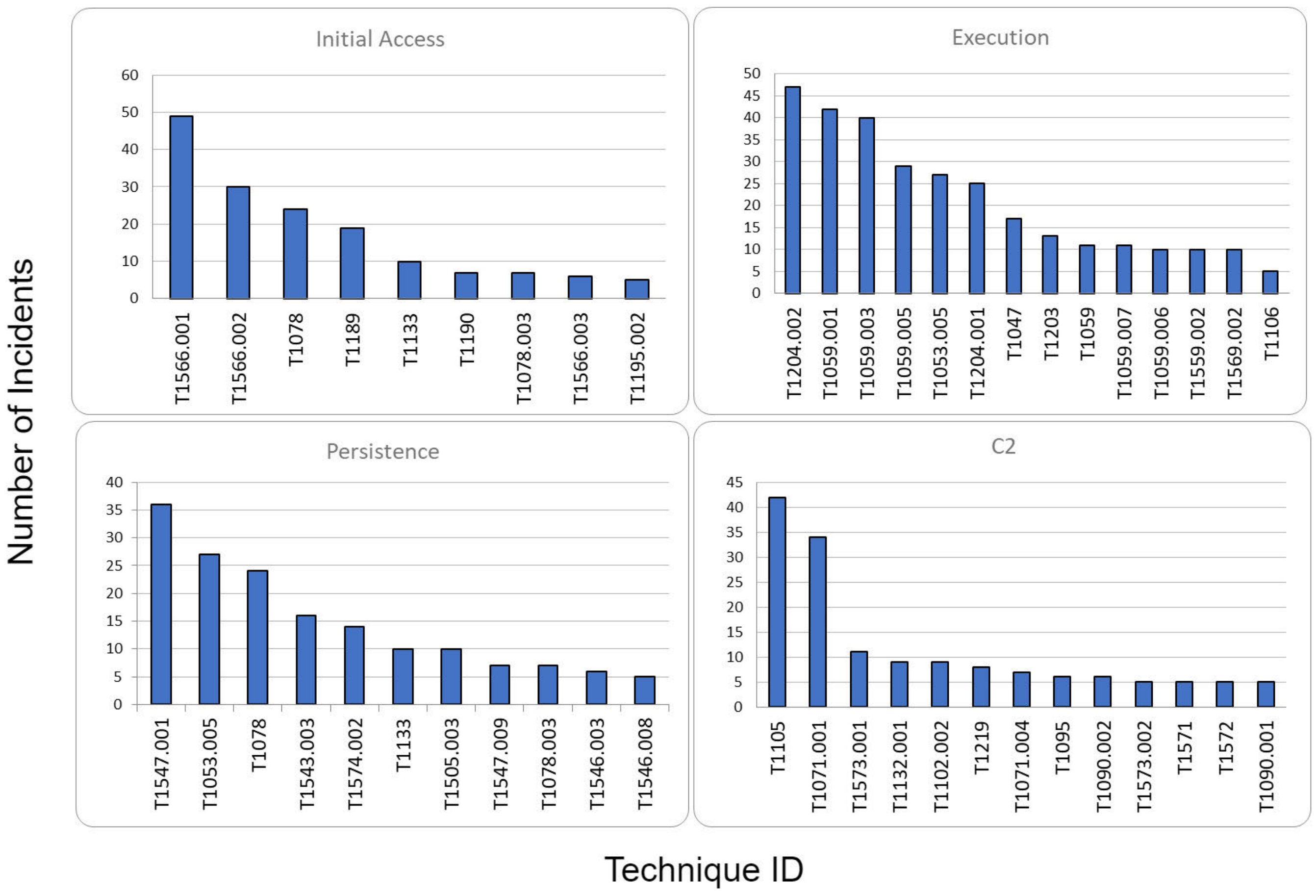

FIGURE 5. Charts of techniques used in incidents reported to MITRE ATT&CK.

cover all those scenarios is 2,089,794. For each COSP, a set of rules should be created. The condition part checks whether there is a correct combination and order of CORIs defined by the COSP. It also checks whether their *Phase* subclass individuals are correlated by determining their *Artifact* subclass individuals correlation via their data properties values. Moreover, the condition part checks whether the *Phase* subclass individuals are subsequent based on the timestamps of their correlated *Artifact* subclass individuals. The action part creates the corresponding *COSP* individual and asserts the *hasPhase* relationship to the forenamed *Phase* subclass individuals. The formation of a *COSP* individual results in the reconstruction and detection of an ongoing cyber-attack.

Since the number of rules for complete coverage is huge, it is suggested that one should start by detecting ongoing cyber-attacks that employ commonly used techniques. This practice of implementing detection based on commonly used techniques first is also encouraged by MITRE [24]. The charts in Figure 5 depict the number of techniques that were used in incidents (i.e., cyber-attacks) reported to MITRE [25]. The assumed y-axis is the number of incidents and the assumed x-axis refers to the ID of the MITRE ATT&CK technique. For instance, the technique T1566.001 was used in 50 incidents to accomplish the Initial Access tactic. Due to space limitations, the techniques that were used a few times (below 5 times) are omitted. As shown in the charts, the commonly used techniques (i.e., the techniques used above 20 incidents) of the Initial Access, Execution, Persistence, and C2 tactics are 3, 6, 3, and 2, respectively. In this vein, one can start implementing Fronesis to detect these commonly used techniques. This will decrease the number of rules that should be developed from over 2 million to 216. In other words, the number of rules that can detect ongoing cyber-attacks that employ commonly used techniques is 18, 54, 36, and 108 for the COSP(DE), COSP(DEI), COSP(DEC), and COSP(DEIC) respectively.

V. EXAMPLE DETECTION

In this Section, the implementation of Fronesis is demonstrated through an email phishing attack example. In particular, it presents how the Fronesis ontology and rule-based reasoning can detect an email phishing attack. Email phishing is a type of social engineering attack where the attacker crafts and sends a fraudulent email in order to deceive the target into acting on behalf of the attacker's benefit [26], [27]. For instance, the target is tricked into downloading a malicious attachment or visiting a malicious URL. Email phishing attack was chosen because of its wide presence in business environments; it is one of the top attack vectors for delivering malware either directly as an attachment or indirectly via a malicious URL [28], [29].

The email phishing attack example spans the Delivery, Exploitation, and Installation phases. These phases are mapped to the Initial Access, Execution, and Persistence tactics as described in subsection 3.2. Each tactic is accomplished via one technique belonging to the tactic. Therefore, the email phishing attack example uses three techniques. One technique for accomplishing each of the forenamed phases. These techniques were selected based on threat reports and the charts presented in Figure 5. The chosen techniques per phase are:

1) Technique for the Delivery phase. According to MITRE ATT&CK, an email phishing attack can start with either the "Spearphishing Link" or "Spearphishing Attachment" or "Spearphishing via Service" techniques. The "Spearphishing Attachment" technique was chosen since malicious attachments are the most used vector of delivering malware through an email phishing attack [30], [31], [32]. The attachment was selected to be an infected Microsoft office.doc file because of its wide presence in email phishing attacks [31], [33].
2) Technique for the Exploitation phase. Based on the charts depicted in Figure 5, the "Malicious File" technique with ID "T1204.002" was chosen. This technique is mostly used for achieving the Execution tactic. It is mapped to the Exploitation phase.
3) Technique for the Installation phase. The "Scheduled Task" technique with ID "T1053.005" was chosen in this example. The Persistence tactic is mapped to the Installation phase.

Based on the above-mentioned techniques, the email phishing attack in our example takes place as follows:

1) Delivery phase. The attacker sends an email to the target. The email contains an attachment.doc file. The target opens the email and downloads the file.
2) Exploitation phase. The target opens the file. A word.exe process runs and opens that file.
3) Installation phase. A scheduled task is created briefly after the file has been opened.

Fronesis was implemented as described in Section 4. Table 1 presents the *Artifact* subclasses along with their axioms that were created for this example. Each axiom follows the OWL Manchester Syntax [34] in the form of <data property name, restriction type, restriction filter>. The data property name is selected in such a manner to convey the attribute of the digital artifact that it represents. For instance, the data property name *hasFullPath* is the *fullpath* attribute of a file. The restriction type quantifies the number of the possible values of the data property. For instance, the *max 1* restriction type quantifies that a *File* can have one *fullpath* attribute at maximum. The restriction filter is the datatype of the data property. Some examples are string and dateTime values. For simplicity, only the necessary information to demonstrate this example is presented.

TABLE 1. Artifact subclasses along with their axioms used in the application example.

Artifact subclass	Axioms
EmailMessage	It is not necessary to include any axiom for the example.
AttachedFile	hasDownloadedTimestamp max 1 xsd:dateTime
File	hasFullPath max 1 xsd:string
Process	hasStartedTimestamp exactly 1 xsd:dateTime
WindowsTask	hasCreatedTimestamp exactly 1 xsd:dateTime

TABLE 2. Object properties between Artifact subclasses used in the example.

Object Property Name	Domain Value	Range Value
hasAttachedFile	EmailMessage	AttachedFile
opensFile	Process	File

Object properties between *Artifact* subclasses are presented in Table 2. The name of each object property (Object Property Name) conveys the meaning of the relationship between two *Artifact* subclasses. One *Artifact* subclass is the domain value of the object property, and the other is its range value. For instance, an EmailMessage *Artifact* subclass can be associated with an AttachedFile *Artifact* subclass using the *hasAttachedFile* relationship (i.e., object property).

As presented in subsection 4.2, there are 4 sets of detection rules. In this example, a COSP(DEI) related rule is illustrated. COSP(DEIC) related rules can be encoded in a similar way and hence are not illustrated here. In our example, there should be at least the following *Artifact* subclass individuals that represent the traces of the email phishing attack in each phase.

- One *EmailMessage* and one *AttachedFile* individual. Both have all properties included in axioms in Table 1 asserted. In addition, the *EmailMessage* individual is associated with the *AttachedFile* individual with the *hasAttachedFile* object property assertion.
- One *Process* and one *File* individual. Both have all properties included in axioms in Table 1 asserted. In addition, the *Process* individual is associated with the File individual with the *opensFile* object property assertion.
- One *WindowsTask* individual that has all properties included in axioms in Table 1 asserted.

Apart from facts about individuals, rule-based reasoning considers declarative rules as detailed in subsection 4.2. The following rules were defined:

1) Rules for recognizing techniques. Each rule checks if there are *Artifact* subclass individuals that represent the traces of a specific technique. In addition, the rule checks if these *Artifact* subclass individuals are related. When both of these conditions are met, the rule creates an individual of the generic OWL Thing class. Finally, the action part of the rule assigns the

TABLE 3. Rules for recognizing techniques.

Technique	Rule
Spearphishing Attachment	EmailMessage(?em) ∧ AttachedFile(?file) ∧ hasAttachedFile(?em, ?file) ∧ swrlx:makeOWLThing(?new, ?em) → Spearphishing_Attachment(?new) ∧ hasTrace(?new, ?em)
Malicious File	Process(?pr) ∧ File(?file) ∧ opensFile(?pr, ?file) ∧ swrlx:makeOWLThing(?new, ?pr) → Malicious_File(?new) ∧ hasTrace(?new, ?pr)
Scheduled Task	WindowsTask(?wtask) ∧ swrlx:makeOWLThing(?new, ?wtask) → Scheduled_Task(?new) ∧ hasTrace(?new, ?wtask)

new individual to the proper *Technique* subclass and asserts the *hasTrace* relation to the *Artifact* subclass individuals. Table 3 presents the rules for recognizing the three techniques used in the email phishing attack example. For instance, the rule that is related to the "Spearphishing Attachment" tries to check if there is an *EmailMessage Artifact* subclass individual which is associated with an *AttachedFile Artifact* subclass individual with the *hasAttachedFile* object property. When this is true, the condition part of the rule creates an individual of the generic OWL Thing class. Afterwards, the action part of the rule assigns the forenamed new individual to the *Spearphishing_Attachment* class and asserts the *hasTrace* relation with the *EmailMessage Artifact* subclass individual. The *AttachedFile Artifact* subclass individual does not have to be associated with the *Spearphishing_Attachment* individual since it is associated with the *EmailMessage Artifact* subclass individual with the *hasAttachedFile* object property. Therefore, it can be retrieved by using this object property.

2) Rules for associating tactics. Each rule is activated if there is a *Technique* subclass individual that can belong to a specific *Tactic* subclass individual. When this condition is met, the rule creates an individual of the generic OWL Thing class. Finally, the action part of the rule assigns the new individual to the proper Tactic subclass and asserts the *hasTechnique* to the *Technique* subclass individual. Table 4 presents the rules for associating tactics. For instance, the rule that is related to the Initial Access tactic is activated when there is a *Spearphishing_Attachment* individual. Then, the condition part of the rule creates a new individual of the OWL Thing class. Finally, the action part assigns the forenamed new individual to the *MA_InitialAccess* class and asserts the *hasTechnique* relation to the *Spearphishing_Attachment* individual.

3) Rules for mapping phases. Each rule is activated if there is a *Tactic* subclass individual that can be mapped to a specific *Phase* subclass individual. When this condition is met, the rule creates an individual of the generic OWL Thing class. Finally, the action part of the rule assigns the new individual to the proper *Phase* subclass and asserts the *mapsTo* relation to the *Tactic* subclass individual. Table 5 presents the rules for mapping *Phase* to *Tactic* subclass individuals. For instance, the rule related to the Delivery phase is activated when there is a *MA_InitialAccess* individual. Then, the rule creates a new individual of the OWL Thing class. Finally, the action part of the rule assigns this new forenamed individual to the *Delivery* class and asserts the *mapsTo* relationship to the *MA_InitialAccess* individual.

TABLE 4. Rules for associating tactics.

Tactic	Rule
Initial Access	Spearphishing_Attachment(?spa) ∧ swrlx:makeOWLThing(?new, ?spa) → MA_InitialAccess(?new) ∧ hasTechnique(?new, ?spa)
Execution	Malicious_File(?mf) ∧ swrlx:makeOWLThing(?new, ?mf) → MA_Execution(?new) ∧ hasTechnique(?new, ?mf)
Persistence	Scheduled_Task(?stasktechnique) ∧ swrlx:makeOWLThing(?new, ?stasktechnique) → MA_Persistence(?new) ∧ hasTechnique(?new, ? stasktechnique)

TABLE 5. Rules for mapping phases.

CKC Phase	Rule
Delivery	MA_InitialAccess(?ia) ∧ swrlx:makeOWLThing(?new, ?ia) → Delivery(?new) ∧ mapsTo(?new, ?ia)
Exploitation	MA_Execution(?exe) ∧ swrlx:makeOWLThing(?new, ?exe) → Exploitation(?new) ∧ mapsTo(?new, ?exe)
Installation	MA_Persistence(?pers) ∧ swrlx:makeOWLThing(?new, ?pers) → Installation(?new) ∧ mapsTo(?new, ?pers)

The last rule needed in this example is the one that detects and creates a *COSP(DEI)*, an ongoing cyber-attack. The rule should conclude if the *Phase* individuals are correlated and subsequent. In our example, this should be performed for the DE, and EI pairs. Two *Phase* subclass individuals are correlated, providing that their traces have attributes with common values. Table 6 presents the DEI rule separated into nine parts which are used to form the overall COSP(DEI) rule as "Part1 ∧ Part2 ∧ Part3 ∧ Part4 ∧ Part5 ∧ Part6 ∧ Part7 ∧ Part8 → Part9".

The Part 1, Part 2 and Part 5 match a CORI. In particular, they match a *Phase* subclass individual, move on to the mapped *Tactic* subclass individual and match the corresponding *Technique* individual. Afterward, they match the *Artifact* subclass individuals, which are the traces of the *Technique* individual and consequently of the *Phase* subclass individual. Finally, they match the properties of the *Artifact* subclass individuals, which are then used in Part 3 and Part 6 to identify if they have common or temporally-related values. When Part 3 and Part 6 are true, the *Delivery*, *Exploitation* and *Installation* individuals are correlated.

Next, Part 4 and Part 7 check if the pairs of the *Delivery* and *Exploitation*, and *Exploitation* and *Installation* Phase individuals are subsequent, respectively. This checking is performed based on the timestamps of their correlated traces to check if the traces of a *Phase* instance are newer than the ones of its preceding *Phase* instance in the pair. The Part 8 creates a new individual of the generic OWL class Thing for

TABLE 6. Declarative rule for detecting the email phishing attack of the application example.

Part	Rule in the Semantic Web Rule Language
1	Delivery(?del) ∧ MA_InitialAccess(?mainit) ∧ mapsTo(?del, ?mainit) ∧ Spearphishing_Attachment(?spa) ∧ hasTechnique(?mainit, ?spa) ∧ EmailMessage(?email) ∧ hasTrace(?spa, ?email) ∧ AttachedFile(?afile) ∧ hasAttachedFile(?email, ?afile) ∧ hasDownloadedTimestamp(?afile, ?downtime) ∧ hasFullPath(?afile, ?afilefullpath)
2	Exploitation(?exploit) ∧ MA_Execution(?exec)∧ mapsTo(?exploit, ?exec) ∧ Malicious_File(?malfile) ∧ hasTechnique(?exec, ?malfile) ∧ Process(?process) ∧ hasTrace(?malfile, ?process) ∧ File(?openedfile) ∧ opensFile(?process, ?openedfile) ∧ hasStartedTimestamp(?process, ?prtime) ∧ hasFullPath(?openedfile, ?openedfullpath)
3	swrlb:equal(?afilefullpath, ?openedfullpath)
4	temporal:before(?downtime, ?prtime)
5	Installation(?instal) ∧ MA_Persistence(?persist) ∧ mapsTo(?instal, ?persist) ∧ Scheduled_Task(?stasktechnique) ∧ hasTechnique(?persist, ?stasktechnique) ∧ WindowsTask(?wtask) ∧ hasTrace(?stasktechnique, ?wtask) ∧ hasCreatedTimestamp(?wtask, ?wtasktime)
6	temporal:duration(?duration, ?wtasktime, ?prtime, "Minutes") ∧ swrlb:lessThanOrEqual(?duration, 1)
7	temporal:before(?prtime, ?wtasktime)
8	swrlx:makeOWLThing(?new, ?del)
9	→ COSP(?new) ∧ hasPhase(?new, ?del) ∧ hasPhase(?new, ?exploit) ∧ hasPhase(?new, ?instal)

TABLE 7. Rules for covering the Registry Run Keys technique.

Used for	Rule
Recognizing technique	WindowsRunKey(?wrkey) ∧ swrlx:makeOWLThing(?new, ?wrkey) → Registry_Run_Keys(?new) ∧ hasTrace(?new, ?wrkey)
Associating tactic	Registry_Run_Keys(?regrunkeys) ∧ swrlx:makeOWLThing(?new, ?regrunkeys) → MA_Persistence ∧ hasTechnique(?new, ?regrunkeys)
Mapping phase	MA_Persistence(?pers) ∧ swrlx:makeOWLThing(?new, ?pers) → Installation(?new) ∧ mapsTo(?new, ?pers)

TABLE 8. Parts 5, 6 and 7 of rule used for detecting an email phishing attack that utilizes Registry Run Keys technique instead of Scheduled Task.

Part	Rule
5	Installation(?instal) ∧ MA_Persistence(?persist) ∧ mapsTo(?instal, ?persist) ∧ Registry_Run_Keys(?regrunkeys) ∧ hasTechnique(?persist, ?regrunkeys) ∧ WindowsRunKey(?wrkey) ∧ hasTrace(?regrunkeys, ?wrkey) ∧ hasCreatedTimestamp(?wrkey, ?wrkeytime)
6	temporal:duration(?duration, ?wrkeytime, ?prtime, "Minutes") ∧ swrlb:lessThanOrEqual(?duration, 1)
7	temporal:before(?prtime, ?wrkeytime)

each *Delivery* individual detected. Lastly, Part 9 is the action part which assigns the new forenamed individual to the *COSP* class and asserts the *hasPhase* relation with the *Delivery*, *Exploitation*, *Installation* individuals.

Figure 6 depicts the output of all the declarative rules performed during the example detection. The COSP individual is depicted at the top of Figure 6 with the name *COSP_1_DEI*. It has three *Phase* individuals, each of which is mapped to a *Tactic* individual. Each *Tactic* individual, in turn, has a *Technique* individual. In the two last rows of boxes, the *Artifact* subclass individuals are depicted. They are the traces of the *Phase* individuals and, therefore, they are the traces of the detected ongoing cyber-attack. The OntoGraf tool, which Figure 6 was based on, does not render that File1 and File2 are the same. However, since the rules resulted in the detection of the spearphishing attack detection, it means that *File1* and *File2* have the same properties and they represent the same digital artifact. Finally, note that the sequence of the *Phase* individuals is in chronological order from left to right since they are subsequent based on the timestamps of their traces.

The example detection includes rules that cover a part of the techniques that can be used to operate an email phishing attack. Tables 3, 4, and 5 can be used as a road-map to create a new rule similar to the one of Table 6 that covers other combinations of techniques. For instance, if the email phishing attack utilized "Registry Run Keys" technique instead of the "Scheduled Task" technique, the rules related to "Scheduled Task" technique in Table 3, 4, and 5 would be replaced by the ones presented in Table 7 while Parts 5, 6 and 7 of Table 6 would be replaced of the ones in Table 8. Note that the trace of the Registry Run Keys technique is an individual of the WindowsRunKey *Artifact* subclass which has a hasCreatedTimestamp axiom.

Performance evaluations were conducted on the implementation of Fronesis based on the email phishing attack of the example detection. A set of 16 rules were created which includes the rule presented in 6. These rules ran against multiple sets of individuals of *Artifact* subclasses with the aim of detecting the email phishing attack. The sets of individuals consisted of 3,000 to 200,000 randomly generated individuals of *Artifact* subclasses. Axioms and object properties in these individuals were also created. Each set of individuals also included the individuals and object properties presented in 1 and 2. Drools rule-engine ran via Protege showed a linear time in detecting the email phishing attack with a mid-level personal computer with 48GB RAM and Intel Core i7-10850H Processor. As depicted in Figure 7, the email phishing attack was detected in 71 seconds and 815 seconds when the 16 rules ran against a set of 10,000 individuals and 200,000 individuals accordingly. This time can still allow for early detection when the digital artifacts of a system are frequently restored, considering the fact that a cyber-attack typically remains undetected for at least 24 days [36].

VI. RELATED WORK

Existing works that are related to Fronesis are rule-based approaches that exploit MITRE ATT&CK or the CKC model for detection purposes. These works are mainly listed in the MITRE's directory in [6]. All of them, especially Sigma and CAR, use rules to detect the operation of techniques within a system. However, they do not focus on the detection of a sequence of techniques that can lead to cyber-attack detection. On the contrary, Fronesis identifies the operation

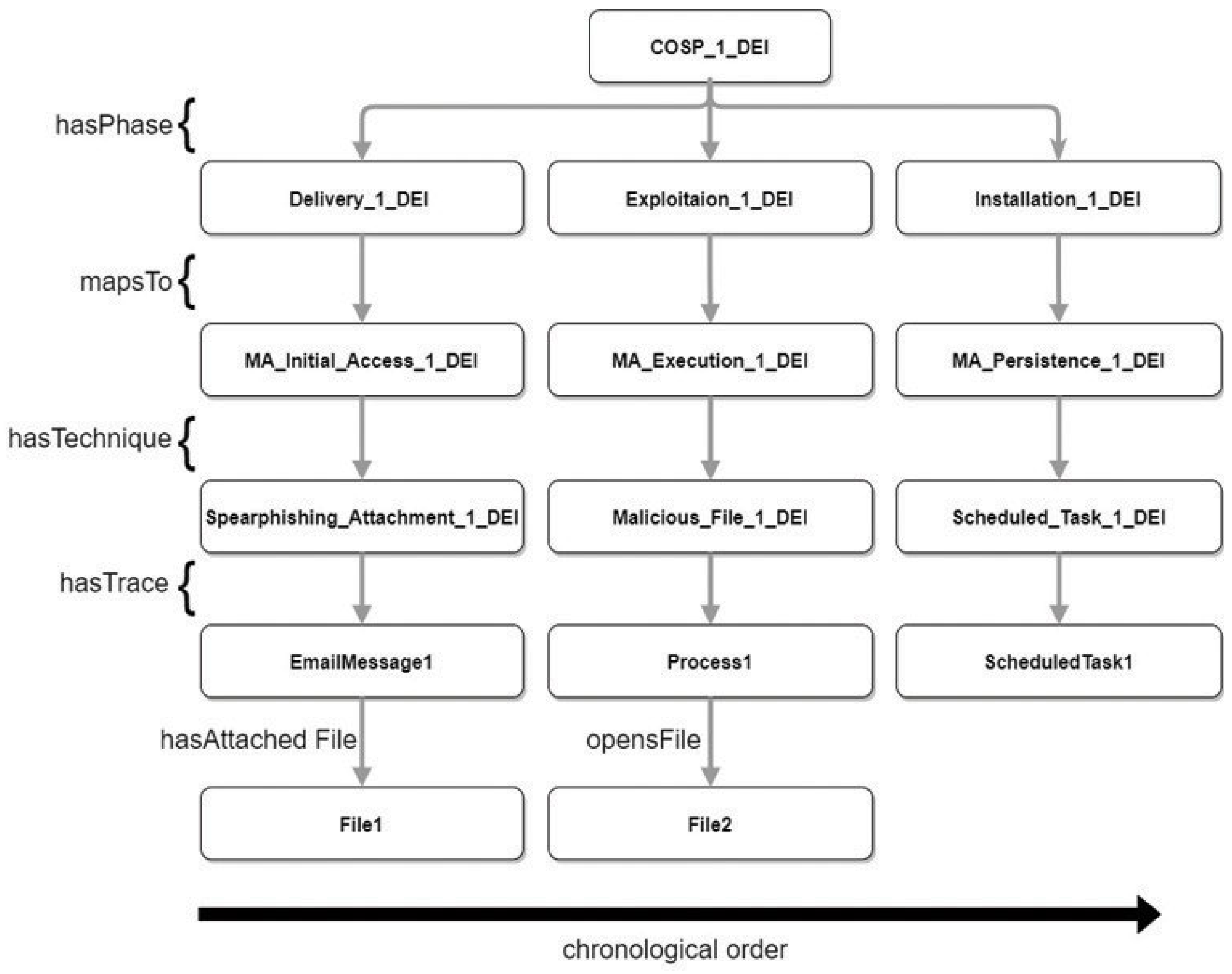

FIGURE 6. **Visualization of a Spearphishing attack detection based on OntoGraf [35].**

of techniques within a system and then utilizes them to detect an ongoing cyber-attack based on the CKC model. Regarding the CAR approach, it does not even consider techniques belonging to the Initial Access tactic. These techniques can be used for detecting initial footholds within a system or network. On the contrary, Fronesis starts the detection process from this tactic since it can be the basis for the early detection of an ongoing cyber-attack. Early detection is crucial for a proper response to be initiated according to the Detect function of the NIST CyberSecurity Framework [1].

The rules of the approaches in [6] are also written in a non-declarative way, especially CAR's rules which are written in pseudocode in order to provide examples of how they can be implemented. On the contrary, Fronesis was implemented with OWL and SWRL rules which are executable and can be checked for inconsistencies. SWRL rules can also be shared among organizations allowing them to build their individual detection capabilities collaboratively in a concise, unambiguous, and declarative way. Finally, the efforts in [6], which are based on Sigma (e.g., Atomic Threat Coverage [37]), including Sigma itself, examines log files (e.g., antivirus, system, and security log files) in their decision logic. In contrast, Fronesis examines digital artifacts which include non-volatile data, such as log files, emails, email attachments, prefetch files, and volatile data, such as running processes. Fronesis, therefore, considers much more information for detection purposes.

Apart from the forenamed efforts, relevant work to Fronesis is [38] which proposes a reasoning process for analyzing incidents (i.e., cyber-attacks) and composing them into campaigns. To do so, the authors propose a logical system, which comprises a model, specific expressions and syntax, that can be used for specifying the reasoning process for incident analysis and campaign composition. The authors applied their proposed logical system in the CKC model to specify the reasoning process for analyzing incidents based

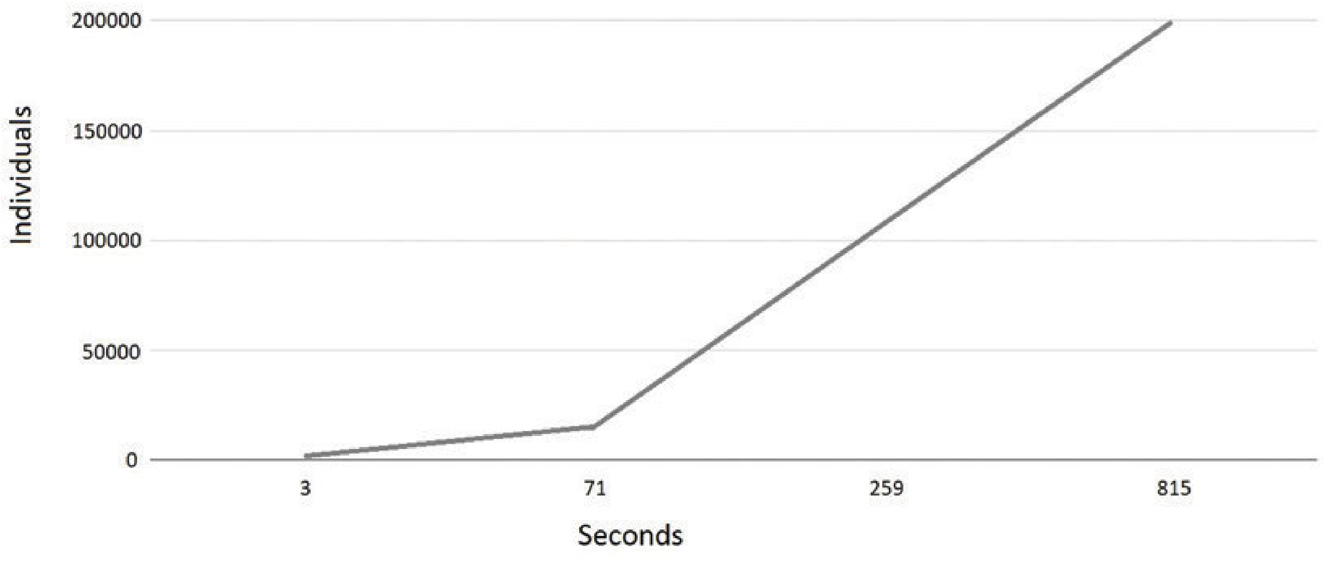

FIGURE 7. Fronesis run time via the Drools rule engine and with 16 declarative rules.

on the CKC model. For this purpose, they specify the pre- and post-conditions for each CKC phase. The post-conditions of one CKC phase should be already observed (i.e., known) so that its pre-conditions can be abducted. These conditions are followed by one in order to analyze an incident by starting from the very last CKC phase (i.e., Actions on Objective) and moving back until all the CKC phases are examined. For example, an Installation's phase post-condition is the installation of a malware in the system in time $t2$ (observed). So, its pre-condition is the execution of the malware in time $t1 < t2$ (abducted). This work, however, does not focus on cyber-detection nor it can be used for. The authors also assume that all CKC phases are accomplished, which might not happen in cases where a CKC phase is skipped in a cyber-attack. Moreover, pre- and post-conditions are high-level recommendations compared to Fronesis that examines digital artifacts ontologically created based on UCO. Finally, the authors do not utilize MITRE ATT&CK to specify the techniques that can occur within a CKC phase. Without knowing the techniques or their traces, one has to examine everything to analyze a cyber-attack. Fronesis, on the other hand, identifies the traces of the operation of each technique which are then mapped to tactics and CKC phases. In this way, Fronesis identifies the traces of each CKC phase. This is detailed in Sections 3.2 and 3.3 via CORIs.

The last effort related to Fronesis is [39], where authors use the CKC model and MITRE ATT&CK to propose a methodology for identifying forensics data, aggregate and correlate them to reconstruct the phases of a cyber-attack. This work, however, is focused on cyber-attack investigation and it needs the cyber-attack as well as its traces to have been detected. In [39] these traces are called forensics data. On the other hand, Fronesis detects an ongoing cyber-attack. The proposed methodology in [39] maps the already detected forensics data to the MITRE ATT&CK techniques manually by following the proof by contradiction approach. One compares these forensics data against the description of MITRE ATT&CK techniques in order to conclude whether they can be the traces of the technique or not. On the contrary, Fronesis is a cyber-attack detection approach that uses rules to identify the operation of each technique automatically by examining digital artifacts. As described in Section 4.2, each rule is written one time when one develops Fronesis based on the description of MITRE ATT&CK techniques. Having the MITRE ATT&CK techniques identified, [39] identifies MITRE ATT&CK tactics and maps them to the CKC model. However, this mapping is different from the one used in Froneses as presented in Section 3. For instance, [39] maps the "initial access" tactic to the "weaponization" CKC phase. Finally, [39] does not consider possible combinations of CKC phases like Fronesis does with COSPs.

TABLE 9. Comparative analysis of Fronesis and related work.

Criteria	Sigma	CAR	[38]	[39]	Fronesis
Individual technique detection	Yes	Yes	No	No	Yes
Sequence of techniques detection	No	No	No	No	Yes
Cyber-attack reconstruction	No	No	Yes	Yes	Yes
Cyber-attack detection	No	No	No	No	Yes
Concise, unambiguous rules	No	No	Yes	Yes	Yes
Digital artifacts consideration	No	No	Yes	Yes	Yes
CKC model consideration	No	No	Yes	Yes	Yes
MITRE ATT&CK consideration	Yes	Yes	Yes	Yes	Yes

Table 9 presents a comparative analysis of the related work presented in this Section against Fronesis based on seven criteria.

VII. CONCLUSION

This paper proposed Fronesis; a digital forensics-based cyber-attack detection approach based on the combined utilization of the MITRE ATT&CK knowledge base, Lockheed Martin's Cyber Kill Chain (CKC) intelligence model, and digital artifacts acquired from the monitored system. Digital artifacts are acquired with proper sensors following digital forensics practices to ensure that the integrity of digital artifacts is preserved. Fronesis examines the digital artifacts in order to recognize MITRE ATT&CK techniques, based on the traces left by the particular procedures of each technique. The recognized techniques are then associated with their MITRE ATT&CK tactics which are mapped to corresponding CKC phases. An ongoing cyber-attack is detected whether the phases are related based on their artifacts and in the correct chronological order. The realization of Fronesis was enabled via an ontology and rules represented respectively in the Web Ontology Language (OWL) and the Semantic Web Rule Language (SWRL) making Fronesis a ruled-based detection approach. The ontology allows for digital artifacts to be represented in an interchangeable and computer processable format, while the rules reason over the facts about artifacts to detect an ongoing cyber-attack. MITRE ATT&CK, CKC, OWL, SWRL, and associated rule-based reasoners are open-source specifications and technologies that would allow for broad adoption of Fronesis.

The proposed detection approach can be implemented as a standalone rule-based detection tool that considers the digital artifacts of the system where it is being operated in order to detect ongoing cyber-attacks. In addition, Fronesis can be integrated into digital forensics tools to support investigations of cyber-attacks. In this case, Fronesis will consider the digital artifacts of a system to identify the presence of a cyber-attack, its traces as well as the utilized MITRE ATT&CK techniques, MITRE ATT&CK tactics, and CKC phases.

Our future research effort will aim at optimizing Fronesis computational performances in order to reduce the time needed to detect an ongoing cyber-attack. Big data technologies, such as Hadoop big data clusters, may be experimented with in this optimization. Our future effort is also focused on the evaluation of Fronesis against cyber-attacks created with the utilization of MITRE Caldera [40]. Finally, the utilization of machine learning (ML) algorithms should be investigated. For instance, Fronesis ontology can be used to define similarity measures for machine learning models that will be used in identifying similarities between digital artifacts. ML algorithm will also be investigated for automatic generation of rules that can be used in ontology-based reasoning to extend Fronesis.

DISCLAIMER

Any mention of commercial products is for information only; it does not imply recommendation or endorsement by NIST.

DECLARATION OF COMPETING INTEREST

The authors declare that they have no financial conflicts of interest.

REFERENCES

[1] M. P. Barrett, "Framework for improving critical infrastructure cybersecurity, version 1.1," NIST Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. CSWP 04162018, Apr. 2018.

[2] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, *Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains*, vol. 1. New York, NY, USA: Academic, 2011.

[3] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.

[4] MANDIANT. *M-Trends 2021: Insights Into Today's Top Cyber Trends and Attacks*. Accessed: Sep. 5, 2021. [Online]. Available: https://www.fireeye.com/current-threats/annual-threat-report/mtrends.html

[5] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "MITRE ATT&CK: Design and philosophy," The Mitre Corporation, McLean, VA, USA, Tech. Rep. 10AOH08A-JC, Mar. 2020.

[6] EU ATT&CK Community. *Directory of ATT&CK Open Source Tools*. Accessed: Mar. 8, 2022. [Online]. Available: https://www.attack-community.org/directory/

[7] K. Kent, S. Chevalier, T. Grance, and H. Dang, "Guide to integrating forensic techniques into incident response," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NIST SP 800-86, Aug. 2006.

[8] S. Alharbi, J. Weber-Jahnke, and I. Traore, "The proactive and reactive digital forensics investigation process: A systematic literature review," in *Information Security and Assurance* (Communications in Computer and Information Science), vol. 200, T.-h. Kim, H. Adeli, R. J. Robles, and M. Balitanas, Eds. Berlin, Germany: Springer, 2011, pp. 87–100.

[9] P. G. Bradford and N. Hu, "A layered approach to insider threat detection and proactive forensics," in *Proc. 21st Annu. Comput. Secur. Appl. Conf. (Technology Blitz)*, 2005.

[10] A. Orebaugh, "Proactive forensics," *J. Digit. Forensic Pract.*, vol. 1, no. 1, pp. 37–41, Mar. 2006.

[11] J. Sachowski, *Implementing Digital Forensic Readiness: From Reactive to Proactive Process*. Boca Raton, FL, USA: CRC Press, 2021.

[12] B. D. Bryant and H. Saiedian, "A novel kill-chain framework for remote security log analysis with SIEM software," *Comput. Secur.*, vol. 67, pp. 198–210, Jun. 2017.

[13] The MITRE Corporation. *MITRE ATT&CK*. Accessed: Sep. 29, 2021. [Online]. Available: https://attack.mitre.org/

[14] V. S. Harichandran, D. Walnycky, I. Baggili, and F. Breitinger, "CuFA: A more formal definition for digital forensic artifacts," *Digit. Invest.*, vol. 18, pp. S125–S137, Aug. 2016.

[15] A. Dimitriadis, "Leveraging digital forensics and information sharing into prevention, incident response, and investigation of cyber threats," Ph.D. dissertation, Dept. Appl. Inform., Univ. Macedonia, Thessaloniki, Greece, 2022.

[16] B. L. Krishna, "Comparative study of fileless ransomware," *Int. J. Trend Sci. Res. Develop.*, vol. 4, pp. 608–616, Apr. 2020.

[17] H. Al-Mohannadi, Q. Mirza, A. Namanya, I. Awan, A. Cullen, and J. Disso, "Cyber-attack modeling analysis techniques: An overview," in *Proc. IEEE 4th Int. Conf. Future Internet Things Cloud Workshops (FiCloudW)*, Vienna, Austria, Aug. 2016, pp. 69–76.

[18] W3C. *OWL 2 Web Ontology Language Document Overview (Second Edition)*. Accessed: Nov. 29, 2021. [Online]. Available: https://www.w3.org/TR/2012/REC-owl2-overview-20121211

[19] W3C. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. Accessed: Nov. 29, 2021. [Online]. Available: https://www.w3.org/Submission/SWRL/

[20] *Drools—Business Rules Management System (Java, Open Source)*. Accessed: Nov. 29, 2021. [Online]. Available: https://www.drools.org/

[21] S. C. for Biomedical Informatics Research at the Stanford University School of Medicine. *Protégé*. Accessed: Nov. 1, 2021. [Online]. Available: https://protege.stanford.edu/

[22] U. Community. *Unified Cyber Ontology (UCO) A Foundation for Standardized Information Representation Across the Cyber Security Domain/Ecosystem*. Accessed: Sep. 9, 2021. [Online]. Available: https://unifiedcyberontology.org/

[23] Clark and P. LLC. *Pellet—Semantic Web Standards*. Accessed: Sep. 10, 2021. [Online]. Available: https://www.w3.org/2001/sw/wiki/Pellet

[24] B. Strom, J. A. Battaglia, M. S. Kemmerer, W. Kupersanin, D. P. Miller, C. Wampler, S. M. Whitley, and R. D. Wolf, ''Finding cyber threats with ATT&CK-based analytics,'' MITRE Mitre Corp., Annapolis Junction, MD, USA, Tech. Rep. MTR170202, Jun. 2017.

[25] MITRE. *Working With ATT&CK | MITRE ATT&CK in Excel*. Accessed: Oct. 31, 2021. [Online]. Available: https://attack.mitre.org/resources/working-with-attack/

[26] S. R. Martin, J. J. Lee, and B. L. Parmar, ''Social distance, trust and getting 'hooked': A phishing expedition,'' *Organizational Behav. Hum. Decis. Processes*, vol. 166, pp. 39–48, Sep. 2019.

[27] T. Halevi, N. Memon, and O. Nov, ''Spear-phishing in the wild: A real-world study of personality, phishing self-efficacy and vulnerability to spear-phishing attacks,'' *SSRN Electron. J.*, Jan. 2015, doi: 10.2139/ssrn.2544742.

[28] *2018 Annual Cybersecurity Report*, Cisco, San Jose, CA, USA, Feb. 2018.

[29] *2021 Threat Report*, BlackBerrry, Waterloo, ON, Canada, 2021.

[30] *Threat Report Q3 2020*, ESET, Bratislava, Slovakia, 2020.

[31] *2021 Email Threat Report*, Trustwave, Chicago, IL, USA, 2021.

[32] C. Beek, M. Cashman, J. Fokker, M. Gaffney, S. Grobman, T. Hux, N. Minihane, L. Munson, C. Palm, T. Polzer, T. Roccia, R. Samani, and C. Schmugar, ''McAfee labs threats report, June 2021,'' McAffee, San Jose, CA, USA, Tech. Rep. 06.21, Jun. 2021.

[33] *2019 Data Breach Investigations Report*, Verizon, New York, NY, USA, May 2019.

[34] W3C. *OWL 2 Web Ontology Language Manchester Syntax (Second Edition)*. Accessed: Dec. 28, 2021. [Online]. Available: https://www.w3.org/TR/owl2-manchester-syntax/

[35] S. Falconer. *OntoGraf*. Accessed: Mar. 10, 2022. [Online]. Available: https://protegewiki.stanford.edu/wiki/OntoGraf

[36] *Today's Top Cyber Trends; Attacks Insights: M-Trends 2021*, Mandiant, Virginia, VA, USA, 2021.

[37] *Atomic Threat Coverage*. Accessed: Mar. 11, 2022. [Online]. Available: https://github.com/atc-project/atomic-threat-coverage

[38] J. M. Spring and D. Pym, ''Towards scientific incident response,'' in *Proc. Int. Conf. Decis. Game Theory Secur.* Seattle, WA, USA: Springer, 2018, pp. 398–417.

[39] C. Liu, A. Singhal, and D. Wijesekera, ''Forensic analysis of advanced persistent threat attacks in cloud environments,'' in *Proc. IFIP Int. Conf. Digit. Forensics*. New Delhi, India: Springer, 2020, pp. 161–180.

[40] *CALDERA*. Accessed: Mar. 16, 2022. [Online]. Available: https://caldera.mitre.org/

ATHANASIOS DIMITRIADIS received the B.Sc. degree from Hellenic Open University, Greece, and the M.Sc. degree in applied informatics and the Ph.D. degree from the University of Macedonia, Greece. He worked as a Digital Forensics Examiner at the public sector in Greece. He is currently a Guest Researcher with the National Institute of Standards and Technology. His research interests include digital forensics and systems integration and interoperability.

EFSTRATIOS LONTZETIDIS received the B.Sc. degree in applied informatics from the University of Macedonia, Greece, and the M.Sc. degree in information security and digital forensics from the University of East London, U.K. He is currently a SoC Analyst at Encode S.A., Greece. His research interests include cyber threat intelligence, digital forensics, and intrusion detection.

BOONSERM KULVATUNYOU received the B.S. degree in mechanical engineering from Chulalongkorn University, Bangkok, Thailand, the M.S. degree from Pennsylvania State University, University Park, PA, USA, and the Ph.D. degree in industrial engineering from Columbia University, New York, NY, USA. He is currently a Researcher with the National Institute of Standards and Technology. His research interests include systems integration and interoperability, particularly for manufacturing systems. He has contributed to several e-business standards. He is currently the Chair of the Industrial Ontology Foundry, Open Applications Group Semantic Refinement Method and Tool Standard Working Group and the IFIP WG 5.7 Smart Manufacturing and Cyber-Physical Production Systems Special Interest Group.

NENAD IVEZIC received the Dipl.Eng. degree from the University of Belgrade, Serbia, and the M.S. and Ph.D. degrees from Carnegie Mellon University. He was a Senior Researcher at the Oak Ridge National Laboratory. He is currently the Group Leader of the Engineering Laboratory, Systems Integration Division,, Process Engineering Group, National Institute of Standards and Technology. He works on measurement science problems that typically lead to testing methods and standards required to solve challenging problems in advanced manufacturing systems. His research interests include systems integration, semantic technologies, standardization, and testing.

DIMITRIS GRITZALIS received the B.Sc. degree in mathematics from the University of Patras, Greece, the M.Sc. degree in computer sciences from the City University of New York, New York, NY, USA, and the Ph.D. degree in information systems security from the University of the Aegean, Greece. He has served as an Associate Rector for Research, the President of the Greek Computer Society, and an Associate Data Protection Commissioner of Greece. He is currently a Professor of cybersecurity with the Department of Informatics, Athens University of Economics and Business, Greece, where he serves as the Director of M.Sc. Program in information systems security and the Director of the INFOSEC Laboratory. His current research interests include situational awareness, critical infrastructure protection, risk assessment, and malware.

IOANNIS MAVRIDIS received the Diploma degree in computer engineering and informatics from the University of Patras, Greece, and the Ph.D. degree in information systems security from the Aristotle University of Thessaloniki, Greece. He is currently a Professor of information security with the Department of Applied Informatics, University of Macedonia, Greece. He also serves as the Director of the Multimedia, Security and Networking (MSN) Laboratory. He has published more than 100 articles in journals and conferences. He is the author and coauthor of three books on information security. He has participated as a principal investigator and a researcher in several international and national funded research and development projects. His current research interests include cybersecurity education, risk management, access control, cyber threat intelligence, digital forensics, and security economics. He serves as an Area Editor (Cybersecurity) of *Array* (Elsevier).