UiO : University of Oslo

Vasileios Mavroeidis

Towards Automated Threat-Informed Cyberspace Defense

Thesis submitted for the degree of Philosophiae Doctor

Department of Informatics
Faculty of Mathematics and Natural Sciences

University of Oslo

2021

Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo
No. 1234

ISSN 1234-5678

Cover: Hanne Baadsgaard Utigard.
Print production: Reprosentralen, University of Oslo.

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of *Philosophiae Doctor* at the University of Oslo. The research presented in this thesis was conducted at the University of Oslo under the supervision of Professor Audun Jøsang and was supported by the Research Council of Norway through grant 247648 for the Oslo Analytics project.

To attain their goals, attackers have established highly automated intelligence-driven attack capability. In contrast, defenders are still challenged by prolonged detection and response times due to their insufficient threat situational awareness and the fact that they heavily rely on manually executed defense processes and procedures.

This thesis comprises four research papers and two standards-track works focused on introducing or enhancing foundational technology in support of accomplishing automated threat-informed cyberspace defense.

The research direction was influenced by the Integrated Adaptive Cyber Defense (IACD) framework that defines three fundamental capability requirements to realize autonomous defense environments that can detect, respond to, or outmaneuver cyber attacks in cyber-relevant time.

Acknowledgements

I would like to express my gratitude to Professor Audun Jøsang for the fruitful discussions and support in this long research journey. In addition, I would like to thank my parents Sophia (Σοφία) and Aristides (Αριστείδης) for their incessant and constant show of love and support. Thank you for always believing in me.

Vasileios Mavroeidis
Oslo, August 2021

List of Papers

Paper I

Vasileios Mavroeidis and Siri Bromander. "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence". In: *Proceedings of the 2017 European Intelligence and Security Informatics Conference (EISIC)*. IEEE, 2017. pp. 91–98. DOI: 10.1109/EISIC.2017.20.

Paper II

Vasileios Mavroeidis and Audun Jøsang. "Data-Driven Threat Hunting Using Sysmon". In: *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy (ICCSP)*. ACM, 2018. pp. 82–88. DOI: 10.1145/3199478.3199490.

Paper III

Vasileios Mavroeidis, Ryan Hohimer, Tim Casey and Audun Jøsang. "Threat Actor Type Inference and Characterization within Cyber Threat Intelligence". In: *Proceedings of the 13th International Conference on Cyber Conflict (CyCon)*. NATO CCDCOE Publications/IEEE, 2021. pp. 327–352.

Paper IV

Vasileios Mavroeidis and Joe Brule. "A Nonproprietary Language for the Command and Control of Cyber Defenses - OpenC2". In: *Computers & Security*. ELSEVIER, 2020. Volume 97, Article 101999. DOI: 10.1016/j.cose.2020.101999.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Transitioning to an entirely interconnected world has introduced new challenges to securing our cyber systems, data, and underlying digital infrastructures. The attack surface of digital assets is proportional to their complexity, functionality, and connectivity. Threat actors have identified that the cyber domain is a flexible territory that can be effectively exploited for a multitude of nefarious reasons, such as conducting illicit activities for profit, interfering with electronic elections by spreading disinformation, performing cyber-vandalism and hacktivism, and engaging in cyber-warfare operations for geopolitical reasons. Adversaries have an asymmetric information and time advantage over defenders and have become increasingly sophisticated and resilient. They make informed decisions regarding their targets, utilize automation, and carefully plan, craft, and execute their attacks to maximize their impact and success rate. On the other hand, defenders are challenged by increased detection and response times due to insufficient threat situational awareness and lack of automation in their cyber operations. To bring this into a perspective, once the adversary gains a foothold inside a target network, *time to compromise* is usually measured in minutes [1]. In contrast, the defenders average *time to detect* is 207 days, and the average *time to contain* is 73 days, for a combined 280 days [3]. Furthermore, the time needed to detect and contain security incidents is directly influencing the cost of a breach. A breach with a lifecycle longer than 200 days on average costs an organization 35% more than one with a shorter lifecycle [3].

To effectively decrease the attack detection and response times, we need to improve our understanding of adversaries and their modus operandi and transition to a more automated, integrated, and adaptive cyber defense. The United States Department of Homeland Security, the National Security Agency's Information Assurance Directorate, and the Johns Hopkins University Applied Physics Laboratory in 2014 introduced the concept of Integrated Adaptive Cyber Defense (IACD)[4]. IACD describes the capability-based requirements for establishing a cyber defense ecosystem that can detect and respond to cyber threats in *cyber-relevant* time. The concept relies on three foundational capabilities [4]. *Automation* of the sensing, sense-making, decision-making, and acting functional blocks of cyber security operations, i.e., Active Cyber Defense (ACD), where different cyber defense systems and components do not operate in isolation, but synergistically at machine speed; *information sharing* about relevant threats for disrupting adversaries from reusing their tools, for anticipatory threat reduction, for making threat-informed decisions, and for performing timely coordinated response operations; and *interoperability* across vendor defense products to

support flexible integration by introducing standardized capability interfaces.

This thesis comprises collected research works that contribute to the attainment of IACD to shorten defenders' attack detection and response times. In particular, we design, develop, test, and integrate technologies that augment or establish the required IACD capability requirements.

1.2 Problem Statement

Influenced by the IACD capability requirements described in Section 1.1, this research identifies three main challenges to address, which are elucidated as *"needs"*.

1.2.1 The Need for Augmented Threat Situational Awareness: Tracking Threat Actor Polymorphism

To increase defenders' threat situational awareness, an IACD environment should consume and utilize Cyber Threat Intelligence (CTI). CTI is actionable and provides relevant, accurate, contextual, and timely knowledge regarding an organization's attack surface, including defensive measures. It provides defenders with the required awareness to decrease their detection and response times by utilizing actionable knowledge within multiple tiers of the overall security operations. Even though defenders have established the required capability to represent and share technical indicators with supplementary context at machine speed by utilizing machine-understandable standards like STIX (Structured Threat Information eXpression)[12], the community still struggles with non-comprehensive, ambiguous, and in many cases, unstructured representations of adversary information, thus, limiting the potential to process, correlate, and analyze that information to understand adversaries better. Another aspect is that adversaries are continually evolving and are becoming hybridized, encompassing multiple motivations and goals, which in turn influence the underlying tactics, techniques, and procedures (TTPs) they utilize and execute in their operations. Defenders need to establish the capability necessary to represent and interpret those new polymorphic threats in a format that both human and machine agents can understand, and use that intelligence to stay threat-informed.

1.2.2 The Need for Automation and Adaptivity in Cyber Defense Operations

To shorten the defenders' detection and response times, IACD requires automating the sensing, sense-making, decision-making, and acting functional areas of cyberspace defense, supported by machine-understandable and -executable codified playbooks (workflows) that impose machine-encoded logic on processes and procedures that human agents would otherwise typically perform.

To be effective, security automation relies heavily on a strong foundation of process documentation, i.e., playbooks. The traditional use of playbooks entails documenting and systematizing an organization's security policies and procedures, such as steps to be performed in response to a triggered condition. For example, playbooks can guide the triaging process to be performed by security analysts to evaluate the relevancy and criticality of an event, or can document standardized incident response processes and ensure the steps are followed in compliance with regulatory frameworks. Playbooks intended to be referenced and performed by humans can be shared and utilized by any organization regardless of its process maturity or automation level. However, it can be challenging to compare playbooks and understand which offer the most suitable models to leverage without having a common documentation template.

A workflow is a machine-readable codification of a playbook to be executed programmatically. A workflow coordinates the interoperation of ACD functional blocks to increase automation in cyber security operations. Orchestration services, otherwise known as Security Orchestration Automation and Response products (SOAR), execute workflows, interfacing with other systems, components, and humans as necessary. Workflows for IACD need to be structured, machine-understandable, adaptive by encoding decision patterns with logic and have the ability to be updated and extended, seamlessly manageable by different orchestration technologies, and shareable. Those requirements demand playbooks and their workflows to be created, documented, codified, and shared in a structured and standardized way across organizational boundaries and technological solutions. Today, playbooks and their codified versions are still defined based on proprietary approaches making them non-portable and challenging to share.

1.2.3 The Need for Standardized Command and Control Interfaces for Interoperability in Integrated Defense

To protect their assets, defenders rely on a plethora of systems and components that operate in isolation and are statically configured, resulting in disintegrated defense environments where they have to serve as integrating (union) blocks.

The next step in cyber defense is to introduce automation and simplify component management by programmatically integrating an organization's existing security solutions and providing the means for centralized command and control. Likewise, this approach has its shortcoming as it requires establishing and maintaining command and control over multiple proprietary product interfaces. An integration is prone to failure every time a vendor updates the application programming interface of a product or when a product is to be introduced, replaced, or removed into the ecosystem.

SOAR is a viable alternative that outsources the complexity of keeping the integrations of an organization's arsenal up-to-date. A user interacts with a proprietary abstraction layer (playbook-driven) that focuses on what it is to achieve, and the platform in the background utilizes its integrations with proprietary product application programming interfaces to perform the desired

activities. As a result, the capability of a SOAR platform is heavily determined by the number of integrations it offers.

A foundational capability requirement for IACD is simplifying product integration by introducing standardized functionality-based command and control interfaces [4]. Introducing standardized function-centric interfaces for command and control can enhance the ability to diversify technologically, makes device management less complicated, and simplifies integration. A standardized language for command and control should be *technology-agnostic* but *function-centric* for *interoperability*; *abstract* so it can be generic enough to be function-centric and can be encoded and transferred via multiple schemes as dictated by the needs of different implementation environments; *minimal* focusing only on the essential information required to derive targeted defense actions; and *extensible* so that it can evolve along with the cyber defense technologies.

1.3 Aim and Research Questions

Even though attackers use automation to execute intelligence-driven attacks, defenders are still challenged with insufficient threat situational awareness and rely heavily on manual cyber defense operations resulting in prolonged detection and response times. The IACD framework was conceptualized to mitigate this asymmetry by promoting the adoption of an extensible, adaptive, commercial off-the-shelf (COTS)-based approach to cyber security operations via automation, interoperability, and information sharing.

This research work aims to contribute to the attainment of IACD by providing enhancements and introducing new foundational technology according to the needs described in Section 1.2. To address the identified needs the following research questions are defined.

- **Research Question 1:** To what extent are the existing semantic ontologies for cyber threat intelligence adequately structured, expressive, and unambiguous to assist the functional areas of sense-making, decision-making, and acting in cyber security operations?
- **Research Question 2:** How can we leverage an ontology to represent and interpret threat actor polymorphism as a way of augmenting defenders' situational awareness against adversaries?
- **Research Question 3:** How can security playbooks be created, documented, and shared in a structured and standardized way across organizational boundaries and technological solutions?
- **Research Question 4:** By using a function-centric approach, how can we standardize the command and control interfaces of cyber defense systems and components to enable a vendor-agnostic plug-and-play capability to product integration?

1.4 Approach and Research Method

In order to answer the research questions defined in Section 1.3, we followed the following approach.

To guide our research direction we identified and investigated the state-of-the-art and relevant literature.

- To determine the level of maturity defenders have established in structuring and representing cyber threat intelligence, we examined existing academic and non-academic works, and performed an evaluation study based on multiple criteria, as defined in Section 2.1 Paper I and [6].
- In order to answer the second research question, we investigated how the existing adversary knowledge bases and other cyber threat intelligence representation approaches capture and interpret persona polymorphism and identified shortcomings in operationalizing them.
- Investigating the existing literature and other available works in support of research questions three and four allowed us to identify and join a newly established technical committee (the year 2017) for standards development, OpenC2, where thereof, we actively participated and led different tasks. In addition, we realized that existing works about documenting and sharing security playbooks and describing adversaries unambiguously in machine-readable formats were insufficient. As a result, we decided to engage in the creation of new standards-track efforts.

We design, implement, and evaluate solutions that address the identified needs in Section 1.2. Where possible, to develop our technologies, we utilized existing established works. For example, to develop the ontology in [5], we extended existing work [2] and utilized it in a different way than its initial purpose.

- To perform the evaluation study in [6], we developed the cyber threat intelligence model and utilized it as a measurement standard to enumerate the expressivity of existing taxonomies, sharing standards, and ontologies within cyber threat intelligence.
- To demonstrate the potentiality of IACD, we developed an integrated and adaptive defense micro-environment that can identify executions of malicious software and automatically issue response actions.
- The need for tracking threat actor polymorphism is addressed by proposing and implementing an ontological solution that encodes domain expertise to automatically infer the types of adversaries based on cyber threat intelligence. Adversaries now can justifiably account for more than one type indicating changes in their behavior at some point in time.

- The need for automation and adaptivity in cyber defense operations is addressed by introducing a common way to document security playbooks (CACAO - Section 2.2.2), a technology that codifies how disparate systems and components can interoperate in response to a triggered condition, positioning humans on the loop in security operations instead of in the loop.
- The need for standardized command and control interfaces in integrated defense is addressed with OpenC2 [7], a standard that commoditizes the command and control interfaces of cyber defense systems and components based on their functionality.

Parts of the work accumulated in this thesis were conducted as part of standards-track tasks within technical committees to ensure high technical quality and consensus across the industry. A technical committee complies with formal processes and procedures for publishing standards defined by the parent Standards Developing Organization (SDO), such as ensuring a product's quality by initiating public review periods to receive feedback. A technical committee conducts qualitative research to capture product requirements and development strategies that will lead to successfully complete a standards-track work and have it embraced by the wider community. In addition, to evaluate the quality and the robustness of technical standards, we organized and participated in plugfests, where we tested the interaction between multiple prototypes and verified interoperability. One of these prototypes was implemented as part of this PhD project.

1.5 Structure of the Thesis

This work is written in the form of a cumulative thesis, compiling four research papers and two standards-track works. The thesis consists of two parts, where Part 1 (Chapters 1 to 3) comprises the motivation for this research, the main research questions, a summary of the research contributions, and a section dedicated to answering this thesis research questions and concluding the research. Part 2 collects the four research papers of this thesis.

Chapter 2

Contributions

2.1 List of Research Papers

Paper I: Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence [6].

This research work identified cyber threat intelligence as a fundamental instrument for increasing defenders' situational awareness about relevant potential threats, **and performed an evaluation study on the conceptual expressivity and availability of existing ontologies, sharing standards, and taxonomies relevant to the task of creating a comprehensive cyber threat intelligence ontology.**

Developing an ontology for cyber threat intelligence requires embracing a modular approach so that we can seamlessly integrate existing domain-relevant ontologies and codified taxonomies. A modular approach to ontology development aids interoperability, provides additional granularity in the development process, and minimizes conceptual breaking changes when a component is removed, added, or replaced.

We developed the Cyber Threat Intelligence model and used it as a measurement standard to conduct part of the evaluation. The Cyber Threat Intelligence model is an abstract model elucidating different types of information that synergistically comprise cyber threat intelligence, such as the who, what, where, when, why, and how of an adversarial operation. The model can also be used as a reference architecture to support the ontological development of concepts for a unified, comprehensive, and extensible cyber threat intelligence ontology.

Using open-source information such as published scientific works, documentation, and source files, we analyzed different taxonomies, sharing standards, and ontologies relevant to cyber threat intelligence. The conducted analysis and evaluation was based on the following criteria:

- Identify information and concepts covered in each work based on the abstraction layers of the Cyber Threat Intelligence model.
- Identify integrations (connections) between ontologies, taxonomies, and cyber threat intelligence sharing schemas for interoperability.
- Characterize the level of comprehensiveness and adequacy of semantic relationships in each work's conceptual layers and recognize the use of logics for information inference.

The study confirmed that little emphasis had been given to developing a comprehensive cyber threat intelligence ontology with existing efforts not being thoroughly designed, being non-interoperable and ambiguous, and lacking semantic reasoning capability.

In particular, barriers to overcome included:

- little focus on dedicated ontological cyber threat intelligence efforts that can account for the strategic, operational, and tactical levels;
- ambiguity in defined concepts which prevents ontology integration and adoption;
- extensive use of prose and limited utilization of existing taxonomies which undermine the querability of the knowledge base and minimize interoperability and the ability to perform reasoning;
- lack of relationships between concepts for augmented cyber threat intelligence interpretation and explainability; and
- minimal use of ontology axioms and constructs that can be used for semantic consistency checking and information inference.

Paper II: Data-Driven Threat Hunting Using Sysmon [8].

This research work demonstrated how structured and unambiguous cyber threat intelligence can support automating different functional areas of cyber defense. Based on the conclusions in Paper I and by referencing the introduced cyber threat intelligence model, we developed an ontology for cyber threat intelligence and orchestrated it to assess the threat level of instantiated software on endpoints. The overall approach utilizes Sysmon agents to receive telemetry from endpoints regarding process execution and other associated behavior. We used formal logic to programmatically infer the threat level of executed processes in a system based on an encoded threat level classification policy that determines the inference requirements. Inference statements can be derived based on standardized security processes such as a sequence of steps to perform triage or include more behavioral and dynamic elements for threat hunting purposes. The comprehensive cyber threat intelligence ontology can not only identify malicious software based on a principled and systematic analysis of log streams but can also be used to increase situational awareness, perform anticipatory threat reduction, and respond to events faster by having access to trusted courses of action.

Paper III: Threat Actor Type Inference and Characterization within Cyber Threat Intelligence [5].

To increase their situational awareness regarding relevant threats, defenders rely on generating, gathering, sharing, and consuming cyber threat intelligence. In that way, one defender's detection becomes another's prevention

by mainly sharing technical artifacts used to inform defense systems. When available, information about the more in-depth understanding of an attack and the attacker is mainly expressed in written prose, is inadequately structured, and is often ambiguous due to the fact that we have not agreed upon standard cyber-threat-intelligence-focused vocabularies. As a result, defenders struggle to machine-operationalize, process, correlate, and analyze different information sources with existing in-house intelligence to better understand adversaries' lifecycle and methods. Adding to that representation challenge, threat actors as part of their evolution process have started to become hybridized, encompassing multiple goals and motivations and operating following formal organizational structures.

This research work reflected the operational and strategic benefits derived from semantically interpreting threat actor intelligence and creating unambiguous and detailed persona profiles based on a standard set of characterization attributes to understand better the actors' nature and capture polymorphism and changes in their behavior and characteristics over time.

We utilized the Threat Agent Library (TAL) [2], which is a set of definitions and descriptions of threat agent categories and their defining attributes, and refined it to create a temporality-based ontological representation.

The domain ontology describes adversaries based on their type, motivations, goals, objectives, visibility, skills, resources and limits, and connects to associated operations for provenance. In addition, the characterization attributes can support creating and executing granular queries over the knowledge base to derive more accurate and relevant intelligence and answer complex questions. Furthermore, universally agreed-upon vocabularies, taxonomies, and definitions enable interoperability and expose interfaces for information fusion. Representing domain knowledge in a declarative form such as axioms and facts can enable automated inference via the ability of machines to reach a conclusion based on evidence. By referencing TAL and using ontology class expressions, we codified the combination of attributes comprising each actor type, allowing a reasoner to act upon the knowledge base and in near real-time infer threat actor types automatically, decreasing cognitive biases manual classification approaches entail.

For exhibiting the potential of our approach, we codified a use-case of a threat actor known to have manifested polymorphic behavior by engaging in operations encompassing multiple diversified goals and motivations. Using our developed ontology, we programmatically inferred the actor's polymorphism and, in particular, its specific associated types demonstrating the actor's behavioral changes over time.

Paper IV: A Nonproprietary Language for the Command and Control of Cyber Defenses - OpenC2 [7].

The complexity and tempo of attacks are constantly increasing, with adversaries utilizing automation to execute them at machine speed. On the other side, defenders operate below their optimal speed with limited automation. To defend their environments, defenders rely on a plethora of technologies that operate in isolation and are statically configured, resulting in poorly integrated cyber security environments and, consequently, resource-demanding defense operations and prolonged detection and response times.

The integration of different cyber defense systems and components can be expensive and requires utilizing and connecting proprietary command and control interfaces. To reduce implementation timelines and cost and increase composability, organizations should be able to seamlessly integrate security products that best suit their needs without requiring connecting custom interfaces and reconfiguring the whole defense ecosystem to interoperate.

This research work presented OpenC2 (Open Command and Control), a suite of specifications that standardizes the command and control interfaces of cyber defense systems and components by adopting a functionality-based approach to device management. Introducing standardized function-centric interfaces for command and control enhances the ability to diversify technologically, makes device management less complicated and simplifies integration.

OpenC2 uses a request-response paradigm where a command is generated and encoded by a producer, transferred to a consumer using a secure transfer protocol, and executed by an actuator.

The OpenC2 specifications can be classified into three different categories.

- The main OpenC2 language specification which provides a lexicon of unambiguous actions and targets for command and control, and defines the proper compositions and data types for the language elements representing the command or response;
- OpenC2 transfer specifications which utilize different existing protocols and standards for encoding and communicating OpenC2 messages securely; and
- OpenC2 actuator profiles which specify subsets of the OpenC2 language and extend it according to particular cyber defense functions. Examples of cyber defense functions include packet filtering, intrusion detection and prevention, and endpoint detection and response.

By enabling a plug-and-play approach to integrated defense, OpenC2 provides vendors with the means to introduce vendor-agnostic interfaces within their technologies portfolio and enable interoperability.

OpenC2 was designed based on the following four principles.

- Technology Agnostic: the OpenC2 language defines a set of unbiased and abstract cyber defense actions for establishing a function-centric approach to command and control and enabling interoperability.
- Concise: an OpenC2 command is designed to be minimal, focusing only on the essential information needed to derive targeted cyber defense actions, and is appropriate for network-constrained environments due to the minimum overhead incurred on the communication channels.
- Abstract: OpenC2 commands and responses are defined abstractly and can be encoded and transferred via multiple schemes as dictated by the needs of different implementation environments.
- Extensible: the OpenC2 language should evolve alongside cyber defense technologies. Supported by the aforementioned design principles, OpenC2 can be extended to introduce new functionality.

The last part of this research work focused on a use case implementation of OpenC2 on a vendor diversified set of packet filters exhibiting standardized function-centric command and control across the different products.

2.2 Standards-Track Work

Contributions to standards described in this section have been produced as part of this PhD project.

2.2.1 Open Command and Control (OpenC2)

The OpenC2 effort is summarized in Section 2.1 and explained in more detail in [7].

2.2.2 Collaborative Automated Course of Action Operations (CACAO) for Cyber Security

Security playbooks document processes and procedures for cyber security and can be used to guide and speed up security operations, ensure organizational policy and regulatory framework compliance, or purely drive automation functions. Thus, security playbooks can be derived in both human-understandable and machine-executable formats. Today, defining security playbooks is based on proprietary templating approaches that prevent programmatic cross-utilization and sharing, making it hard for users to compare generic playbooks and understand which offer the best models to leverage.

The Collaborative Automated Course of Action Operations (CACAO) Security Playbooks is a standards-track work [11] that defines a playbook schema and taxonomy for the purpose of standardizing

the way we create, document, and share security playbooks. A CACAO playbook is a workflow for security orchestration containing a set of steps to perform based on a logical process and may be triggered by an automated or manual event or observation. In other words, playbooks are the driving force of integrated defense, guiding systems, subsystems, and human agents on how to interoperate to execute a course of action. At a high-level, playbooks comprise a set of workflow steps that utilize logic to control the commands to be executed or performed, a set of commands to execute or perform, and targets that accept, receive, process, or execute the commands. A CACAO playbook, among other command types, can encapsulate OpenC2 commands and utilize the interoperability provided at the actuator level; thus, making CACAO playbooks requiring minimal modifications to map to an organization's own environment.

CACAO defines two playbook classes.

- Executable: an executable playbook is intended to be immediately actionable in an organization's security infrastructure without requiring modifying or updating the workflow and commands.
- Template: a playbook template provides reference actions related to a particular security incident, malware, vulnerability, or other security operation. A template playbook will not be immediately executable by a receiving organization.

CACAO currently [10] defines the following playbook types.

- Notification playbook: a notification playbook primarily focuses on the orchestration steps required to notify and disseminate information and other playbooks about a security event, incident, or other threat. For example, a notification playbook can be used to notify multiple entities about an attack and disseminate other playbooks to detect and mitigate it as quickly as possible.
- Detection playbook: a detection playbook primarily focuses on the orchestration steps required to detect a known security event, other known or expected security-relevant activity, or for threat hunting.
- Investigation playbook: an investigation playbook primarily focuses on the orchestration steps required to investigate what a security event, incident, or other security-relevant activity has caused. Investigation playbooks will likely inform other subsequent actions upon completion of the investigation.
- Prevention playbook: a prevention playbook primarily focuses on the orchestration steps required to prevent a known or expected security event, incident, or threat from occurring. Prevention playbooks are often designed and deployed as part of best practices to safeguard organizations from known and perceived threats and behaviors associated with suspicious activity.

- Mitigation playbook: a mitigation playbook primarily focuses on the orchestration steps required to mitigate a security event or incident that has occurred when remediation is not initially possible. Mitigation playbooks are designed to reduce or limit the impact of suspicious or confirmed malicious activity. For example, a mitigation playbook can be used to quarantine affected users, devices, or applications from the network temporarily to prevent additional problems. Mitigation usually precedes remediation, after which the mitigation actions are reversed.

- Remediation playbook: a remediation playbook primarily focuses on the orchestration steps required to remediate, resolve, or fix the resultant state of a security event or incident, and return the system, device, or network back to a nominal operating state. Remediation playbooks can fix affected assets by selectively correcting problems due to malicious activity by reverting the system or network to a known good state.

- Attack playbook: an attack playbook primarily focuses on the orchestration steps required to execute a penetration test or attack simulation to test or verify security controls or identify vulnerabilities within an organization's environment.

Furthermore, CACAO supports threat intelligence sharing efforts by providing a common way to express and incorporate courses of action within intelligence feeds, providing the knowledge required to quickly detect or respond to cyber attacks. Defenders utilizing security orchestration, automation, and response technologies (SOAR) can benefit from shared playbooks that can programmatically be translated or natively consumed by their tool.

Chapter 3

Conclusion

This thesis supports the attainment of Integrated Adaptive Cyber Defense (IACD) by providing contributions that establish or enhance the framework's capability requirements, aiming to decrease the attack detection and response times of defenders and disrupt adversaries from reusing their tools.

In particular, the capability requirement to disrupt adversaries from reusing their tools and techniques is supported by increasing defenders' situational awareness via rapid threat intelligence sharing using standards like STIX [12] and by referencing contextual knowledge bases like MITRE ATT&CK [9] to provide the information needed to better prepare against known threats and techniques. To enhance defenders' knowledge about adversaries, we introduced an ontological approach for representing and modeling their personas based on their defining characteristics. Furthermore, using the reasoning property of ontologies, we codified domain expertise to automatically inferring adversary types. We also remarked on the importance of agreeing upon and standardizing well-defined vocabularies for unambiguously enriching threat actor context (e.g., motivations, types, objectives, capability, role in operations) and easier processing, correlating, and analyzing information from different sources. In addition, we introduced CACAO security playbooks, a standard for documenting courses of action that can be utilized in combination with cyber threat intelligence to disseminate an effective and comprehensive coordinated response plan within an intelligence or IACD community.

The capability requirement to automate the sensing, sense-making, decision-making and acting functions to provide network defense in cyber-relevant time is supported by security orchestration. Security orchestration integrates and guides the synergistic operation of systems and components in a defense ecosystem based on documented processes and procedures. CACAO playbooks (see Section 2.2.2) is a standard for creating, documenting, and sharing security-related processes and procedures. Due to their standardized nature, CACAO playbooks are portable, allowing defenders to share, exchange, and utilize them.

The capability requirement to establish standardized command and control interfaces for interoperability so that different tools can integrate without requiring pairwise custom interfaces is achieved with OpenC2 [7, 13]. OpenC2 standardizes the acting portion of cyber defense by introducing a function-centric approach to command and control. OpenC2 is vendor-agnostic and enables a plug-and-play approach to product integration. CACAO playbooks that utilize OpenC2 increase playbook portability by minimizing the re-configurations needed when being transferred to different consumer technological environments.

The rest of the chapter revisits the research questions formulated in Section 1.3 and discusses them in connection with the contributions of this thesis.

3.1 Summary of Contributions

3.1.1 Research Question 1: To what extent are the existing semantic ontologies for cyber threat intelligence adequately structured, expressive, and unambiguous to assist the functional areas of sense-making, decision-making, and acting in cyber security operations?

Paper I identified that little emphasis has been given to developing a semantic web ontology dedicated to representing cyber threat intelligence, with existing efforts being non-thoroughly designed and non-modular for describing different concepts, inadequately structured and ambiguous, and lacking reasoning capability that could introduce a form of automation in knowledge inference. Following up on those challenges, Paper II elucidated how a modular and comprehensive cyber threat intelligence ontology can integrate into cyberspace defense operations and support and influence the sense-making, decision-making, and acting functional blocks. The ontology was based on existing threat-intelligence-relevant efforts to aid interoperability, like for correlating information from multiple sources, and was utilized to demonstrate an active cyber defense capability by automatically inferring the threat level of instantiated software on endpoints and programmatically deciding on courses of action.

3.1.2 Research Question 2: How can we leverage an ontology to represent and interpret threat actor polymorphism as a way of augmenting defenders' situational awareness against adversaries?

Based on the research presented in Paper III, an ontology is a viable solution for aggregating, structuring and representing, and correlating knowledge about adversaries in order to understand them better and recognize polymorphism. An ontology provides a structured approach to knowledge representation and presents interfaces for both manual and programmatic analysis. Using codified statements, a reasoner can programmatically infer new information, such as distinguishing how adversaries evolve into new behaviors and inferring their types. In addition, to support organizing what is known and to achieve a level of automation in information inference, it is evident that we need to eliminate ambiguity and inconsistencies across different threat intelligence-relevant information sources by introducing and utilizing standard contextual vocabularies for enrichment.

3.1.3 Research Question 3: How can security playbooks be created, documented, and shared in a structured and standardized way across organizational boundaries and technological solutions?

CACAO (presented in Section 2.2.2) is an open-source specification that standardizes the way we create, document, and share cyber security playbooks. CACAO security playbooks are both human- and machine-readable and have been designed to be vendor-agnostic, allowing them to be transferable from one technological environment to another (interoperability) with minimal modifications. Defenders can utilize CACAO playbooks to drive, inform, and speed up their prevention, detection, investigation, and response capabilities or to validate the adequacy of the existing security controls and preparedness against threats, always by using orchestration. The standardized and open-source nature of CACAO also benefits threat intelligence sharing by providing a common way to describe courses of action.

3.1.4 Research Question 4: By using a function-centric approach, how can we standardize the command and control interfaces of cyber defense systems and components to enable a vendor-agnostic plug-and-play capability to product integration?

Paper IV presented Open Command and Control (OpenC2), a suite of specifications that, based on a function-centric approach, standardizes the way cyber defense systems are managed. OpenC2 aligns with the capability requirement of the IACD framework for interoperability by introducing a vendor-agnostic plug-and-play approach to product integration. Systems and components within a cyber defense ecosystem can be introduced, replaced, and managed without requiring connecting custom interfaces and reconfiguring the whole defense ecosystem to interoperate. Actuator profiles unambiguously aggregate common functionality among cyber defense systems and components of the same function and provide a standard language for their command and control.

Bibliography

[1] Ahlberg, C. *The Threat Intelligence Handbook: Moving Toward a Security Intelligence Program*. CyberEdge Group, 2019.

[2] Casey, T. *Threat Agent Library Helps Identify Information Security Risks*. Tech. rep. Intel Corporation, 2007.

[3] *Cost of a Data Breach Report*. Tech. rep. Ponemon Institute and IBM Security, 2020.

[4] Done, B. K. et al. "Towards a Capability-Based Architecture for Cyberspace Defense". In: Concept Paper Approved for Public Release, US Department of Homeland Security, US National Security Agency Information Assurance Directorate, and the Johns Hopkins University Applied Physics Laboratory, AOS-16-0099. 2016.

[5] Mavroeidis, V. et al. "Threat Actor Type Inference and Characterization within Cyber Threat Intelligence". In: *Proceedings of the 2021 13th International Conference on Cyber Conflict (CyCon)*. 2021.

[6] Mavroeidis, V. and Bromander, S. "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence". In: *Proceedings of the 2017 European Intelligence and Security Informatics Conference (EISIC)*. IEEE. 2017, pp. 91–98.

[7] Mavroeidis, V. and Brule, J. "A Nonproprietary Language for the Command and Control of Cyber Defenses – OpenC2". In: *Computers & Security* vol. 97 (2020).

[8] Mavroeidis, V. and Jøsang, A. "Data-Driven Threat Hunting Using Sysmon". In: *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy*. 2018, pp. 82–88.

[9] MITRE Corporation. *Adversarial Tactics, Techniques Common Knowledge (ATT&CK)*. Accessed: Jan. 2021. [Online]. Available: https://attack.mitre.org/.

[10] OASIS. *CACAO Security Playbooks Version 1.0*. OASIS Committee Specification 02. OASIS, 2021.

[11] OASIS. *Collaborative Automated Course of Action Operations (CACAO)*. Accessed: Jul. 2021. [Online]. Available: https://www.oasis-open.org/committees/cacao/.

[12] OASIS. *OASIS Structured Threat Information Expression (STIX™) Version 2.1*. Standard. OASIS, 2021.

[13] OASIS. *Open Command and Control (OpenC2)*. Accessed: Jan. 2021. [Online]. Available: https://www.oasis-open.org/committees/openc2/.

Papers

Paper I

Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence

Vasileios Mavroeidis[1], Siri Bromander[2]

Revision 01
Date: February 2021
Editor: Vasileios Mavroeidis

This paper is an updated version of DOI: 10.1109/EISIC.2017.20 and includes language enhancements. The changes in no case have affected the paper's scope, analysis, and derived conclusions.

Originally published in: *Proceedings of the 2017 European Intelligence and Security Informatics Conference (EISIC)*, Athens, Greece, September 2017, pp. 91–98, DOI: 10.1109/EISIC.2017.20, IEEE.

[1]University of Oslo, Oslo, Norway, vasileim@ifi.uio.no
[2]mnemonic, Oslo, Norway, siri@mnemonic.no

Abstract

Cyber threat intelligence is the provision of evidence-based knowledge about existing or emerging threats. Benefits of threat intelligence include increased situational awareness and efficiency in security operations and improved prevention, detection, and response capabilities. To process, analyze, and correlate vast amounts of threat information and derive highly contextual intelligence that can be shared and consumed in meaningful times requires utilizing machine-understandable knowledge representation formats that embed the industry-required expressivity and are unambiguous. To a large extend, this is achieved by technologies like ontologies, interoperability schemas, and taxonomies. This research evaluates existing cyber-threat-intelligence-relevant ontologies, sharing standards, and taxonomies for the purpose of measuring their high-level conceptual expressivity with regards to the who, what, why, where, when, and how elements of an adversarial attack in addition to courses of action and technical indicators. The results confirmed that little emphasis has been given to developing a comprehensive cyber threat intelligence ontology with existing efforts not being thoroughly designed, non-interoperable and ambiguous, and lacking semantic reasoning capability.

I.1 Introduction

Defenders utilize multiple diversified defense products to prevent, detect, and disrupt incoming attacks. However, the increasing capability, persistence, and complexity of adversarial attacks have made traditional defense approaches ineffective.

Organized cybercrime is at a peak. PwC's global economic crime survey of 2016 [28] reports that there are organizations that have suffered cybercrime losses over $5 million, and of these, nearly a third reported losses over $100 million. Juniper Research [33] reports that cybercrime will increase the cost of data breaches to $2.1 trillion globally by 2019, four times the estimated cost of breaches in 2015.

For enhancing their security posture, defenders recognized the need to understand threats their organization may face better and started exchanging threat information aiming one organization's detection to become another's prevention. This practice has achieved a certain maturity, with organizations focusing on generating and sharing more contextual and robust information known as cyber threat intelligence. Organizations rely on cyber threat intelligence to identify and understand impending attacks, speed up security operations, and drive and prioritize the implementation of security controls.

Threat intelligence is referred to as the task of gathering evidence-based knowledge, including context, mechanisms, indicators, implications and actionable advice, about an existing or emerging menace or hazard to assets that can be used to inform decisions regarding the subject's response to that menace

or hazard[1]. Cyber threat intelligence needs to be relevant, timely, accurate, actionable, and contextual.

To a large extent, intelligence generation, consumption, and interpretation should be automated processes that leverage machine-understandable representation formats that allow for scalable processing, correlation, and analysis. A type of knowledge representation is ontologies. Ontologies encode knowledge about a particular domain in a structured manner, leverage logic for performing inference, and are flexible and modular, allowing them to be easily extended, refined, or interconnect with other ontologies.

Working towards an ontology for cyber threat intelligence has its challenges. Our research reports the following as the largest barriers to overcome:

- little focus on dedicated ontological cyber threat intelligence efforts that can account for the strategic, operational, and tactical levels;
- ambiguity in defined concepts that prevents ontology integration and adoption;
- extensive use of prose and limited utilization of existing taxonomies that undermine the querability of the knowledge base and minimize interoperability and the ability to perform reasoning;
- lack of relationships between concepts for augmented cyber threat intelligence interpretation and explainability; and
- minimal use of ontology axioms and constructs that can be used for semantic consistency checking and information inference.

This article evaluates taxonomies, sharing standards, and ontologies relevant to the task of creating a comprehensive cyber threat intelligence ontology. To achieve that, we created the cyber threat intelligence model that indicates different types of information as abstraction layers that all together elucidate a malicious attack's five W's and one H; who, what, why, where, when, how, and technical indicators. We pinpoint the mappings between the cyber threat intelligence model and the taxonomies, sharing standards, and ontologies evaluated, aiming to indicate their expressivity. Finally, we critically review the shortcomings of the current cyber threat intelligence ontology approaches, and we discuss various directions to improving their quality.

I.2 Methodology

This section introduces two models related to threat detection maturity and cyber threat intelligence. The two models overlap, and both can meet different needs that are explained in the next two subsequent subsections. The Cyber Threat Intelligence model is the basis of the evaluation process conducted in this research.

[1] https://www.gartner.com/doc/2487216/definition-threat-intelligence

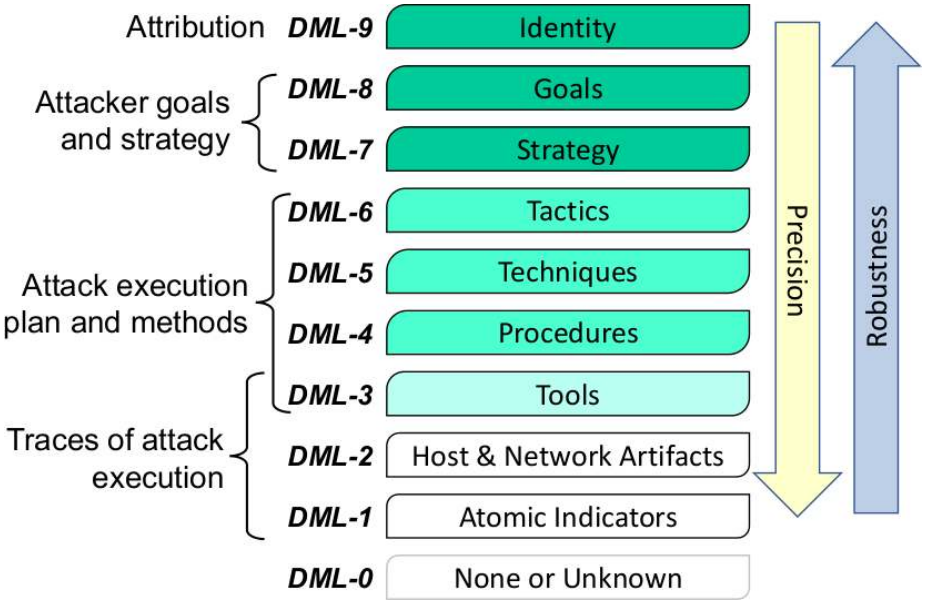

Figure I.1: Modified Detection Maturity Level Model

I.2.1 The Detection Maturity Level Model - DML

Ryan Stillions proposed the Detection Maturity Level (DML) model in 2014 [35]. DML is used to describe an organization's maturity regarding its ability to consume and act upon given cyber threat intelligence (Figure I.1). Detection maturity at the higher levels of DML indicates that an organization has established intelligence-driven processes and procedures for detecting, understanding, and responding to cyber threats more effectively and efficiently. In 2016, we extended this model by adding an additional level (9) "Identity" and presented it for use in the semantic representation of cyber threats [3].

I.2.2 The Cyber Threat Intelligence Model

The Cyber Threat Intelligence model builds upon and extends [35] and [3], and intends to elucidate the different types of information an organization needs access to increase its situational awareness about threats. In this research, we utilize our model as a measurement standard. We use the model's distinguished cyber threat intelligence abstraction layers to measure the expressivity of existing taxonomies, sharing standards, and ontologies.

The remaining section is devoted to specifying the definitions of the elements comprising the cyber threat intelligence model.

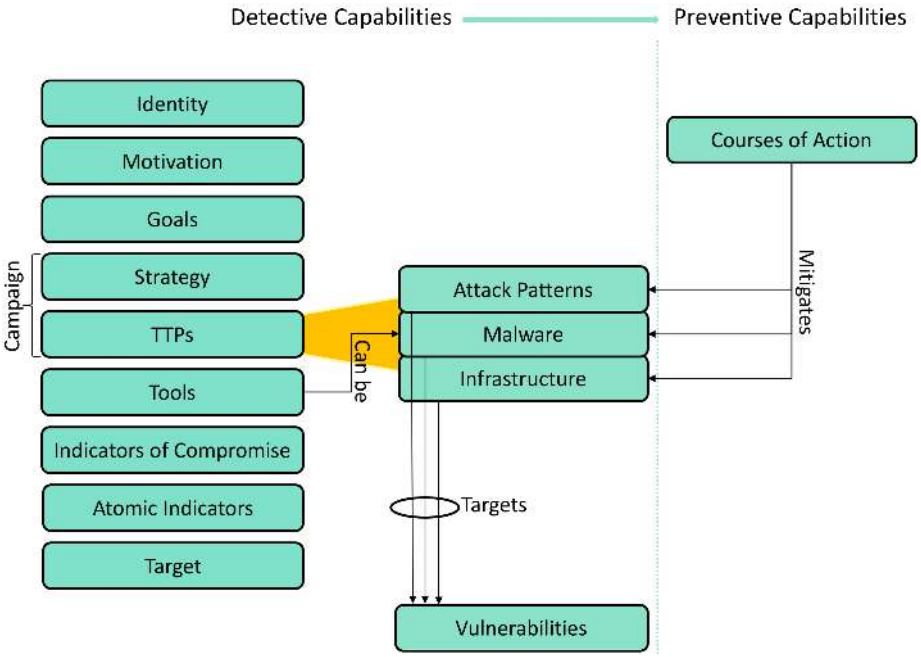

Figure I.2: Cyber Threat Intelligence Model

Identity: the identity of a threat actor can be the real name of a person, an organization, a group's affiliates, or a nation-state-backed entity. In cases that attribution is not feasible, tracking operations via persona-based threat actor profiles also has its benefits as it allows identification of an actor's behavioral characteristics concerning their motivations, goals, capability, and TTPs they utilize.

Motivation: can be described as the driving force that enables actions to pursue specific goals. The goals of an attacker may change, but the motivation most of the times remains the same. Knowing a threat actor's motivation narrows down which targets that actor may focus on, helps defenders focus their limited defensive resources on the most likely attack scenarios, as well as shapes the intensity and the persistence of an attack [5]. Examples of motivation can be ideological, geopolitical, and financial.

Goals: according to Fishbach and Ferguson [8] "a goal is a cognitive representation of a desired endpoint that impacts evaluations, emotions, and behaviors". A goal consists of an overall end state and the behavior objects and plans needed for attaining it. The activation of a goal guides behaviors. Depending on how the attack is organized, the goal might not be known for the attacking team executing the attack. The team might only receive a strategy to follow. In current cyber threat intelligence approaches and knowledge bases goals are mostly described in prose. A goal can be defined as a tuple of two:

(Action, Object), but work needs to be done to create a consistent taxonomy at an adequate level of detail [3]. Typical examples of goals are "steal intellectual property", "damage infrastructure", and "embarrass a competitor".

Strategy: is a non-technical high-level description of the planned attack. There are typically multiple ways an attacker can achieve its goals, and the strategy defines which approach the threat agent should follow.

TTPs: tactics, techniques, and procedures are aimed to be consumed by a more technical audience. TTPs characterize adversary behavior in terms of what they want to achieve technically and how they are doing it.

Attack Pattern: is a type of TTP that describes behavior attackers use to carry out their attacks.

Malware: is a type of TTP and refers to a software that is inserted into a system with the intent of compromising the target in terms of confidentiality, integrity, or availability.

Infrastructure: describes any systems, software services, and any associated physical or virtual resources intended to support an adversarial operation, such as using purchased domains to support Command and Control, malware delivery sites, and phishing sites.

Tools: attackers install and use tools within the victim's network. Tools encompass both dedicated software developed for malicious reasons and software intended for different use (e.g., vulnerability and network scanning, remote process execution) but utilized for malicious purposes, mainly for avoiding detection (defense evasion).

Indicators of Compromise: are actionable technical elements and are directly consumable by cyber defense systems and components for detecting malicious or suspicious activity. A good IOC encompasses contextual information in addition to behavioral, computed, or atomic indicators to assist situational awareness.

Atomic Indicators: the value of atomic indicators is limited due to their short shelf life. Atomic indicators include file hashes, domain names, and IPs.

Target: represents the entity an attack is directed to and can be an organization, a sector, a nation, or individuals.

Course of Action: refers to measures that can be taken to prevent or respond to attacks.

I.2.3 Evaluation Criteria

Using open-source information such as related publications, documentation, or source files, the next section of the article presents and analyzes different taxonomies, sharing standards, and ontologies relevant to cyber threat intelligence. The conducted analysis/evaluation is based on the following criteria:

- Identify information and concepts covered in each work based on the abstraction layers of the Cyber Threat Intelligence model (Figure I.2). Table 1 presents the results.

- Identify integrations (connections) between ontologies, taxonomies, and cyber threat intelligence sharing schemas for interoperability (Sections III, IV).

- Characterize the level of comprehensiveness and adequacy of semantic relationships in each work's conceptual layers and recognize the use of logics for information inference (Sections IV, V).

A number of identified articles present ontologies that are not described in great detail and have no reference to the actual ontology files (RDF/OWL), making their evaluation hard to achieve. Furthermore, some available ontology efforts do not offer an additional supporting publication and, most of the times, not even proper documentation.

I.3 Taxonomies and Sharing Standards

This section provides an overview of taxonomies and sharing standards that are used or potentially can be used in cyber threat intelligence representation. We categorize them as enumerations, scoring systems, and sharing standards.

I.3.1 Enumerations

TAL (Threat Agent Library) [6] is a set of standardized definitions and descriptions to represent significant threat agents. The library does not represent individual threat actors, thus it is not intended to identify people, or investigating actual security events. The goal of TAL is to help in risk management and specifically to identify threat agents relevant to specific assets. In that way, security professionals pro-actively can build defenses for specific threats.

Casey, in 2015, introduced a new taxonomy for cyberthreat motivations. The taxonomy identifies drivers that cause threat actors to commit illegal acts [5]. Knowing these drivers could indicate the nature of the expected harmful actions.

CVE (Common Vulnerabilities and Exposures) [17] is a list of records for publicly known information-security vulnerabilities in software packages.

NVD (National Vulnerability Database) [23] is a repository of standards-based vulnerability management data represented using the Security Content Automation Protocol (SCAP). The NVD performs analysis on CVEs that have been published to the CVE dictionary. This analysis results in association impact metrics (Common Vulnerability Scoring System - CVSS), vulnerability types (Common Weakness Enumeration - CWE), and applicability statements (Common Platform Enumeration - CPE), as well as other pertinent metadata. This data enables automation of vulnerability management, security measurement, and compliance.

CPE (Common Platform Enumeration) [16] is both a specification and a list. The specification defines standardized machine-readable methods for assigning and encoding names to IT product classes (software and hardware). The CPE dictionary provides an agreed-upon list of official CPE names.

CWE (Common Weakness Enumeration) [18] is a dictionary of software and hardware security weaknesses aiming to enhance understanding about common flaws and their mitigation.

CAPEC (Common Attack Patterns Enumerations and Characteristics) [15] provides a collection of the most common attack methods used to exploit known weaknesses.

ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) [14] is focused on network defense and describes the operational phases in an adversary's lifecycle, pre- and post-exploit, and details the TTPs adversaries use to achieve their objectives while targeting, compromising, and operating inside a network. It is a valuable resource to understand better adversary behavior and can be used for multiple purposes, such as for adversary emulation, behavioral analytics, cyber threat intelligence enrichment, defense gap assessment, and red teaming and SOC maturity assessment. ATT&CK matrices exist about adversary behavior targeting enterprise environments, mobile and industrial control systems. Moreover, information pertinent to software adversaries' use, mitigation techniques, procedure examples, and detection recommendations are also available.

I.3.2 Scoring Systems

CVSS (Common Vulnerability Scoring System) [22] is a measurement standard aiming to score vulnerabilities based on their severity. Combined with timely CTI, CVSS can inform an organization about which vulnerability remediation activities should prioritize.

CWSS (Common Weakness Scoring System) [19] is part of CWE and it provides a mechanism for scoring weaknesses using 18 different factors. It is worth mentioning that Mitre's Common Weakness Risk Analysis Framework (CWRAF) can be used in conjunction with CWSS to identify the most important CWEs applying to a particular business and their deployed technologies. The difference between CVSS and CWSS is that the first one targets specific software vulnerabilities scoring, whereas the latter one targets CWE scoring.

I.3.3 Sharing Standards

A study of existing threat intelligence sharing initiatives concluded that Structured Threat Information eXpression (STIX) is currently the most used standard for sharing threat information [31]. STIX is an expressive, flexible, and extensible representation language used to communicate an overall piece of threat information [1]. The STIX architecture comprises different cyber threat information elements such as cyber observables, indicators, incidents, adversaries tactics, techniques, procedures, exploit targets, courses of action, cyber attack campaigns, and threat actors. Furthermore, STIX was recently redesigned and as a result omits some of the objects and properties defined in the first version. The objects chosen for inclusion in the second version represent a minimally viable product that fulfills basic consumer and producer requirements for cyber

threat intelligence sharing. Both standards can be used and adapted based on an organization's needs. It is worth pointing out that MITRE additionally offers MAEC (Malware Attribute Enumeration and Characterization) [20], a very expressive malware sharing language for encoding and communicating high-fidelity information about malware based upon attributes such as behaviors, artifacts, and attack patterns. MAEC can be integrated in STIX or used as a standalone.

OpenIOC, developed by Mandiant, is an extensible XML schema that enables you to describe the technical characteristics that identify a known threat, an attacker's methodology, or other evidence of compromise. The types of information covered directly by OpenIOC are derived mainly by enriched low-level atomic indicators, comprising indicators of compromise, thus covering the IOC category of the cyber threat intelligence model.

I.4 Ontologies

Since the work of Blanco et al. [2] in 2008, we have not found any overviews of existing ontologies within the cyber security domain. The authors remark that the scientific community has not accomplished a general security ontology because most of the works are focused on specific domains or the semantic web. The same conclusion was drawn by Fenz and Ekelhart [7]. Additionally, Blanco et al. [2] emphasize the complication of combining their identified ontologies due to the non-common interpretation and different terms applied to similar concepts in different ontologies. Our study confirms the same almost 10 years after the study of Blanco et al. [2].

While several ontologies relevant to the broader cybersecurity domain exist, only a small number was identified relating to threat information and threat intelligence representation. For a number of them, identifying the mappings to the abstraction layers of the cyber threat intelligence model is challenging because they are described only at a very high level and without having any relevant RDF/OWL files available for further investigation. The ontologies analyzed hereafter are listed chronologically based on their publication date.

Stefan Fenz and Andreas Ekelhat [7] described an information security ontology that can be used to support a broad range of information security risk management methodologies. The high-level concepts of the ontology are derived from the security relationship model described in the National Institute of Standards and Technology Special Publication 800-12. Concepts to represent the information security domain knowledge include threat, vulnerability, control, attribute, and rating. In addition, concepts such as asset, organization, and person are necessary to formally describe organizations and their assets. Lastly, the concept of location is integrated combined with a probability rating concept to interrelate location and threat information in order to assign priory threat probabilities. Like other works, the authors have difficulties connecting ambiguous concepts deriving from different standards (e.g., ambiguous distinction between a threat and a vulnerability).

Wang and Guo [39] proposed an ontology for vulnerability management and analysis (OVM) populated with all existing vulnerabilities in NVD. The basis of the ontology is built on the results of CVE and its related standards, such as CWE, CPE, CVSS, and CAPEC. OVM captures the relationships between the following concepts which constitute the top level of the ontology; vulnerability, introduction phase (software development life cycle - time periods during which the vulnerability can be introduced), active location (location of the software where the flaw manifests), IT product, IT vendor, product category (such as web browsers, application servers, etc.), attack (integration of CAPEC), attack intent, attack method, attacker (human being or software agent), consequence, and countermeasure.

Obrst et al. [25] suggested a methodology for creating an ontology based on already well-defined ontologies that can be used as modular sub-ontologies. In addition, they remark the usefulness of existing schemas, dictionaries, glossaries, and standards as a form of knowledge acquisition of the domain by identifying and analyzing entities, relationships, properties, attributes, and range of values that can be used in defining an ontology. Their suggested ontology is based on the diamond model of malicious activity [4], which expresses the relationships between an adversary (actor), the capabilities of the adversary, the infrastructure or resources the adversary utilizes, and the target of the adversary (victim). The authors state that they developed first the aspects of infrastructure and capabilities, but they are still not in the level of detail they desire. In addition, their current ontology is focused on malware and some preliminary aspects of the diamond model.

A good argumentation for transitioning from taxonomies to ontologies for intrusion detection was made in 2003, by Undercoffer et al. [37]. They suggested an ontology that would enable distributed anomaly-based host IDS sensors to contribute to a common knowledge-base, which again would enable them to detect quicker a possible attack.

Based on this, More et al. [21] in 2012, suggested to build a knowledge-base with reasoning capabilities to take advantage of an extended variety of heterogeneous data sources, to be able to identify threats and vulnerabilities. Their data sources suggest that data retrieved and included in the ontology is within the atomic indicators category of the CTI model.

Oltramari et al. [26] proposed a three-layer cyber security ontology named "CRATELO" aiming at improving the situational awareness of security analysts, resulting in optimal operational decisions through semantic representation. Following the methodology of [25], the authors build upon existing ontologies and extend them. Specifically, CRATELO includes the top-level ontology DOLCE-SPRAY extended with security-related middle-level ontology (SECCO) capable of capturing details of domain specific scenarios, such as threat, vulnerability, attack, countermeasure, and asset. The low-level sub-ontology, cyber operations (OSCO), is the extension of the middle-level ontology.

Gregio et al. [11] suggested an ontology to address the detection of modern complex malware families whose infections involve sets of multiple exploit methods. To achieve this, they created a hierarchy of main behaviors each

one of them consisting of a set of suspicious activities. Then they proposed an ontology that models the knowledge on malware behavior. They state that a given program behaves suspiciously if it presents one or more of the six events (main behaviors) described below which consist of several characteristics. The events are attack launching, evasion, remote control, self-defense, stealing, and subversion. When new set of process actions with malicious behaviors appear (input from "transformed" log files), the ontology can be inferred to see if an instance of suspicious execution is linked to a malware sample.

Salem and Wacek [30] designed a data extraction tool called TAPIO (Targeted Attack Premonition using Integrated Operational data), specializing in extracting data (through the use of natural language processing) and automatically mapping that data into a fully linked semantic graph accessible in real-time. Part of TAPIO is a cybersecurity ontology known as Integrated Cyber Analysis System (ICAS) that ingests extracted data (logs and events) from several sources to provide relationships across an enterprise network. The tool aims to help incident response teams connect and correlate events and actions into an ontology for automatic interpretation. ICAS is a collection of 30 sub-ontologies specializing in specific conceptual areas as part of host-based and network-based conceptual models.

Iannacone et al. [12] described their STUCCO ontology, which is developed to work on top of a knowledge graph database. The STUCCO ontology design is based upon scenarios of use by both human and automated users and incorporates data from 13 different structured data sources with different format. The data included in the current STUCCO ontology fall into the categories identity, TTPs, tools, and atomic indicators of our cyber threat intelligence model. Their future work included extending the ontology to support STIX.

Gregio, Bonacin, de Marchi, Nabuco, and de Geus [10] extended the work of Gregio et al. [11] and introduced the malicious behavior ontology (MBO). MBO is capable of detecting modern complex malware families whose infections involve sets of multiple exploit methods, by applying SWRL rules to the ontology for inferencing. In addition, these rules also apply metrics to specify whether a program is behaving maliciously or not and specifically, how suspicious the execution of a program is. The authors state that their model is able to detect unknown malicious programs even in cases where traditional security mechanisms like antivirus are not, by performing automatic inference of suspicious executions in monitored target systems. However, the current state of the ontology has some limitations such as performance issues, cannot detect malware in real time, and false positives and negatives. Based on its operation MBO can provide useful indicators of compromise for malware.

Fusun et al. suggested ontologies like attacks, systems, defenses, missions, and metrics for quantifying attack surfaces [9]. Their Attack Surface Reasoning (ASR) gives a cyber defender the possibility to explore trade-offs between cost and security when deciding on their cyber defense composition. ASR is mainly modeled after the Microsoft STRIDE [32] threat classification framework, which categorizes attack steps into 6 categories and is to the extent of our knowledge not the preferred framework within threat intelligence community due to its

lack of details. In comparison, CAPEC and CPE have around 500 and 1000 categories, respectively.

As part of their study on using security metrics for security modeling, Pendelton et al. suggested the Security Metric Ontology [27]. The ontology includes four sub-ontologies; vulnerability, attack, situations and defense mechanisms, and describes the relationship between them. The terminology used is somewhat different than that of known taxonomies, and their aim at modeling metrics is more prominent than that of analysis and reasoning. The ontology is published on GitHub[2].

The Unified Cybersecurity Ontology was suggested by Syed et al. [36] in 2016. It serves as a backbone for linking cyber security and other relevant ontologies. There are mappings to aspects of STIX, and references to CVE, CCE, CVSS, CAPEC, STUCCO and KillChain. The concepts are loosely connected at a very high level and the lack of OWL constructs decreases the reasoning capabilities of the ontology. In addition, our analysis indicate that the use of domain and range restrictions would result in faulty classifications when used with a reasoner. The ontology is published on GitHub[3].

The Unified Cyber Ontology has been introduced on GitHub[4], without any academic publications to date and no actual RDF/OWL files yet. The model ontology is however interesting as it originates from the creators of STIX, which is currently the most used format for sharing threat intelligence[31]. The content of that work is driven primarily by the initial base requirements of expressing cyber investigation information and is the product of input from the Cyber-investigation Analysis Standard Expression community (CASE)[5].

Without any publication, we find the Cyber Intelligence Ontology (CIO), published only on GitHub[6] to be relevant. This GitHub repository includes most of the mentioned taxonomies and sharing standards in this article, encoded in OWL. The limitation of those ontologies is that they are not connected or unified. For the aforementioned reason, we do not include CIO in the analysis and the evaluation table.

I.5 Discussion

Intelligence-driven defense augments organizations' detecting and responding capabilities and introduces a more informed preventive approach to the overall cybersecurity operations. The maturity, the analytical skills, and the available information sources of a security team determine their capability to produce accurate and actionable threat intelligence [29][13].

To leverage the benefits of ontologies and description logics in cyber threat intelligence, we need unambiguous representations with sufficient expressivity

[2]https://github.com/marcusp46/security-metrics-ontology
[3]https://github.com/Ebiquity/Unified-Cybersecurity-Ontology
[4]https://github.com/ucoProject/uco
[5]https://github.com/casework/case
[6]https://github.com/daedafusion/cyber-ontology

and robust explicable bindings between concepts. A reference architecture like the one provided by our Cyber Threat Intelligence model can be used as an engineering blueprint that can support the fundamental development of concepts through modular domain ontologies that can be the basis for establishing a bigger and more comprehensive ontology that is extensible and adaptive. The analysis of the existing ontological efforts confirmed that there is still a small focus and much work to be done to establish a comprehensive and unambiguous cyber threat intelligence ontology.

Ontologies are modular and extensible, allowing replacing or integrating with other domain-focused ontologies to build a more holistic one that can benefit from an augmented representation regarding a domain of interest. In the ontologies evaluated, we identified that the lack of OWL expressions is a common phenomenon. Expressions make ontologies powerful by encoding domain expertise for reasoning. Using the encoded knowledge, a reasoner can infer new information from the existing asserted information at machine speed, introducing a form of automation.

Furthermore, we cannot ignore mentioning the limited taxonomy encodings and integrations we observed and the missing interconnections between those taxonomies and existing ontologies for establishing more standardized (interoperability) and expressive representations that resolve ambiguity, like taxonomies that standardize threat actor motivations, goals, and types. The importance of standardizing and utilizing taxonomies is apparent in cases where higher querability levels are desired.

Overall, an ontology gives access to a knowledge base containing rich historical and present information in a robust, meaningful, and explicable way. Analysts can utilize a cyber threat intelligence ontology to perform analytical tasks while decreasing the confirmation biases entailed in purely manual analytical and decision-making processes.

I.6 Conclusion

Our study concluded that there is much work to achieve before establishing a contextual and unambiguous cyber threat intelligence ontology. Barriers to overcome include little focus on dedicated ontological cyber threat intelligence efforts that can account for the strategic, operational, and tactical levels; ambiguity in ontology concepts that prevent ontology integration and adoption; extensive use of prose and limited utilization of existing taxonomies that undermine the querability of the knowledge base and the ability to perform reasoning; lack of relationships between concepts that can support interpretation and explainability; and minimal use of ontology axioms and constructs that can be used for semantic consistency checking and information inference.

Acknowledgements. This research was supported by the research projects Oslo Analytics (Grant No. 247648), TOCSA (Grant No. 263375), and ACT (Grant No. 256785) funded by the Research Council of Norway.

Table I.1: Evaluation of Taxonomies, Sharing Standards, and Ontologies

		Identity	Motivation	Goal	Strategy	TTP	Tool	IOC	Atomic Indicator	Target	COA
Taxonomies	TAL [6]	*									
	Threat Agent Motivation [5]	*	*								
	CVE [17]								*		
	NVD [23]								*		
	CPE [16]								*		
	CWE [18]					*			*		*
	CAPEC [15]					*		*			*
	ATT&CK [14]	*				*	*				
	CVSS [22]								*		
	CWSS [19]								*		
Sharing Standards	STIX 1 [1]	*	*	* (Intended Effect:taxonomy)	*	*	*	*	*	*	*
	STIX 2 [24]	*	*	* (Objectives:string)	*	*	*	*	*	*	*
	MAEC [20]							*			
	OpenIOC [34]					*	*	*	*		
Ontologies	Fenz & Ekelhat (2009) [7]								*		
	Wang & Guo (2009) - OVM [39]					*			*		*
	Orlet et al. (2012) [25]	*					*		*	*	
	More et al. (2012) [21]					*			*		
	Oltramari et al. (2014) - CRATELO [26]	*				*			*	*	
	Gregio et al. (2014) [11]						* (malware)		*		
	Salem & Wacek (2015) - ICAS [30]					*			*		
	Iannacone et al. (2015) - STUCCO [12]	*				*	*		*		
	Gregio et al. (2016) - MBO [10]						* (malware)	*	* (it may provide)		
	Fusun et al. (2015) - ASR [9]					*			*	*	
	Pendleton et al. (2016) - Security Metrics Ontology [27]					*					*
	Syed et al. (2016) - UCO [36]	*	*	*	*	*	*	*	*		*
	Unified Cyber Ontology (2016) - UCO [38]	*	*	*	*	*	*	*	*	*	*

References

[1] Barnum, S. *Standardizing Cyber Threat Intelligence Information with the Structured Threat Information eXpression (STIX ™)*. Tech. rep. MITRE Corporation, 2012.

[2] Blanco, C. et al. "A Systematic Review and Comparison of Security Ontologies". In: *Third International Conference on Availability, Reliability and Security (ARES)*. IEEE. 2008, pp. 813–820.

[3] Bromander, S., Jøsang, A., and Eian, M. "Semantic Cyberthreat Modelling". In: *Proceedings of the 11th International Conference on Semantic Technology for Intelligence, Defense, and Security (STIDS)*. 2016, pp. 74–78.

[4] Caltagirone, S., Pendergast, A., and Betz, C. *The Diamond Model of Intrusion Analysis*. Tech. rep. DTIC Document, 2013.

[5] Casey, T. *Understanding Cyber Threat Motivations to Improve Defense*. Tech. rep. Intel Corporation, 2015.

[6] Casey, T. *Threat Agent Library Helps Identify Information Security Risks*. Tech. rep. Intel Corporation, 2007.

[7] Fenz, S. and Ekelhart, A. "Formalizing Information Security Knowledge". In: *Proceedings of the 4th International Symposium on information, Computer, and Communications Security*. ACM. 2009, pp. 183–194.

[8] Fishbach, A. and Ferguson, M. J. "The Goal Construct in Social Psychology". In: *Social Psychology: Handbook of Basic Principles* (2007), pp. 490–515.

[9] Fusun, M. B. et al. "Using Ontologies to Quantify Attack Surfaces". In: *Proceedings of the 11th International Conference on Semantic Technology for Intelligence, Defense, and Security (STIDS)*. 2016.

[10] Grégio, A. et al. "An Ontology of Suspicious Software Behavior". In: *Applied Ontology* vol. 11, no. 1 (2016), pp. 29–49.

[11] Grégio, A. et al. "Ontology for Malware Behavior: A Core Model Proposal". In: *Proceedings of the 23rd IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*. IEEE. 2014, pp. 453–458.

[12] Iannacone, M. et al. "Developing an Ontology for Cyber Security Knowledge Graphs". In: *Proceedings of the 10th Annual Cyber and Information Security Research Conference*. ACM. 2015.

[13] Johnson, C. et al. *Guide to Cyber Threat Information Sharing*. NIST Special Publication. NIST, 2016.

[14] MITRE Corporation. *Adversarial Tactics, Techniques and Common Knowledge*. [Online]. Available: https://attack.mitre.org/.

[15] MITRE Corporation. *Common Attack Pattern Enumeration and Classification*. [Online]. Available: https://capec.mitre.org/.

[16] MITRE Corporation. *Common Platform Enumeration*. [Online]. Available: https://cpe.mitre.org/specification/.

[17] MITRE Corporation. *Common Vulnerabilities and Exposures*. [Online]. Available: https://cve.mitre.org.

[18] MITRE Corporation. *Common Weakness Enumeration*. [Online]. Available: https://cwe.mitre.org.

[19] MITRE Corporation. *Common Weakness Scoring System*. [Online]. Available: https://cwe.mitre.org/cwss/cwss_v1.0.1.html.

[20] MITRE Corporation. *Malware Attribute Enumeration and Characterization*. [Online]. Available: https://maec.mitre.org.

[21] More, S. et al. "A Knowledge-Based Approach to Intrusion Detection Modeling". In: *Proceedings of the 2012 IEEE Symposium on Security and Privacy Workshops (SPW)*. IEEE. 2012, pp. 75–81.

[22] NIST. *Common Vulnerability Scoring System*. [Online]. Available: https://nvd.nist.gov/vuln-metrics/cvss/.

[23] NIST. *National Vulnerability Database*. [Online]. Available: https://nvd.nist.gov/.

[24] OASIS CTI TC. *Structured Threat Information Expression (STIX ™) 2.0*. [Online]. Available: https://oasis-open.github.io/cti-documentation/. 2017.

[25] Obrst, L., Chase, P., and Markeloff, R. "Developing an Ontology of the Cyber Security Domain". In: *Proceedings of the 7th International Conference on Semantic Technology for Intelligence, Defense, and Security (STIDS)*. 2012, pp. 49–56.

[26] Oltramari, A. et al. "Building an Ontology of Cyber Security". In: *Proceedings of the 9th International Conference on Semantic Technology for Intelligence, Defense, and Security (STIDS)*. Citeseer. 2014, pp. 54–61.

[27] Pendleton, M. et al. "A Survey on Systems Security Metrics". In: *ACM Comput. Surv.* vol. 49, no. 4 (Dec. 2016).

[28] PwC. *Global Economic Crime Survey 2016*. Tech. rep. PwC, 2016.

[29] Rid, T. and Buchanan, B. "Attributing Cyber Attacks". In: *Journal of Strategic Studies* vol. 38, no. 1-2 (2015), pp. 4–37.

[30] Salem, M. B. and Wacek, C. "Enabling New Technologies for Cyber Security Defense with the ICAS Cyber Security Ontology". In: *Proceedings of the 10th International Conference on Semantic Technology for Intelligence, Defense, and Security (STIDS)*. 2015, pp. 42–49.

[31] Sauerwein, C. et al. "Threat Intelligence Sharing Platforms: An Exploratory Study of Software Vendors and Research Perspectives". In: *Proceedings of the 13th Internationale Tagung Wirtschaftsinformatik (WI)*. Ed. by Leimeister, J. M. and Brenner, W. 2017.

[32] Shostack, A. *Threat Modeling: Designing for Security*. John Wiley & Sons, 2014.

[33] Smith, S. *Cybercrime will Cost Businesses over $2 Trillion by 2019*. Accessed: Jun. 2017. [Online]. Available: https://www.juniperresearch.com/press/press-releases/cybercrime-cost-businesses-over-2trillion/.

[34] *Sophisticated Indicators for the Modern Threat Landscape: An Introduction to OpenIOC*. Tech. rep. Mandiant Corporation, 2013.

[35] Stillions, R. *The DML Model*. Accessed: Sep. 2016. [Online]. Available: https://ryanstillions.blogspot.com/2014/04/the-dml-model_21.html. Blog Post. 2014.

[36] Syed, Z. et al. "UCO: A Unified Cybersecurity Ontology". In: *Proceedings of the AAAI Workshop on Artificial Intelligence for Cyber Security*. AAAI Press. 2016.

[37] Undercoffer, J., Joshi, A., and Pinkston, J. "Modeling Computer Attacks: An Ontology for Intrusion Detection". In: *Proceedings of the International Workshop on Recent Advances in Intrusion Detection*. Springer. 2003, pp. 113–135.

[38] *Unified Cyber Ontology*. [Online]. Available: https://github.com/ucoProject/uco. GitHub Repository. 2016.

[39] Wang, J. A. and Guo, M. "OVM: An Ontology for Vulnerability Management". In: *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*. ACM. 2009.

Paper II

Data-Driven Threat Hunting Using Sysmon

Vasileios Mavroeidis[1], Audun Jøsang[2]

Revision 01
Date: March 2021
Editor: Vasileios Mavroeidis

This paper is an updated version of DOI: 10.1145/3199478.3199490 and includes language enhancements. The changes in no case have affected the paper's scope, analysis, and derived conclusions.

Originally published in: *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy (ICCSP 2018)*, Guiyang, China, March 2018, pp. 82–88, DOI: 10.1145/3199478.3199490, ACM.

[1]University of Oslo, Oslo, Norway, vasileim@ifi.uio.no
[2]University of Oslo, Oslo, Norway, audun.josang@mn.uio.no

Abstract

Threat actors can be persistent, motivated and agile, and leverage a diversified and extensive set of tactics and techniques to attain their goals. In response to that, defenders establish threat intelligence programs to stay threat-informed and lower risk. Actionable threat intelligence is integrated into security information and event management systems (SIEM) or is accessed via more dedicated tools like threat intelligence platforms. A threat intelligence platform gives access to contextual threat information by aggregating, processing, correlating, and analyzing real-time data and information from multiple sources, and in many cases, it provides centralized analysis and reporting of an organization's security events. Sysmon logs is a data source that has received considerable attention for endpoint visibility. Approaches for threat detection using Sysmon have been proposed, mainly focusing on search engine technologies like NoSQL database systems. This paper demonstrates one of the many use cases of Sysmon and cyber threat intelligence. In particular, we present a threat assessment system that relies on a cyber threat intelligence ontology to automatically classify executed software into different threat levels by analyzing Sysmon log streams. The presented system and approach augments cyber defensive capabilities through situational awareness, prediction, and automated courses of action.

II.1 Introduction

Utilizing threat intelligence has become a priority in cybersecurity operations as a way to prevent an attack or decrease the time needed to discover and respond to an attack. In addition, cyber-attacks are increasingly sophisticated, posing significant challenges for organizations that must defend their data and systems from capable threat actors. Threat actors can be persistent, motivated, and agile, and they use multiple tactics, techniques, and procedures to disrupt the confidentially, integrity and availability of systems and data. Given the risks of the present cyber threat landscape, it is essential for organizations to focus on utilizing cyber threat intelligence and participate in threat information sharing to improve their security posture. In previous work, [4], we discussed the importance of having access to cyber threat intelligence for increased situational awareness and presented the Cyber Threat Intelligence model that enables cyber defenders to explore their threat intelligence capability and understand their position against the ever-changing cyber threat landscape. Furthermore, in the same work, we commented on the importance of developing a multi-layered comprehensive cyber threat intelligence ontology for improving the threat detection, prioritization, and response capabilities of organizations. The results of [4] indicated that little emphasis had been given to developing a comprehensive cyber threat intelligence ontology, although some holistic initiatives toward that goal existed [2, 8, 9].

Threat detection and analysis requires aggregating logs into a centralized system known as security information and event management (SIEM). A SIEM

collects logs by deploying multiple collection agents that gather security-related events from endpoints, servers, and other security systems and appliances to perform analysis and detect unwanted behavior. In particular, one resource that has received attention for endpoint visibility is Sysmon, a Windows system service and device driver that monitors and logs system activity of Windows workstations. Proposed approaches for threat detection using Sysmon mainly focus on search engines (NoSQL database systems) or graph databases. Without any relevant academic publication, a comprehensive list of related works can be found on GitHub[1].

The contribution of this paper is twofold. First, we present a comprehensive Cyber Threat Intelligence Ontology (CTIO), based on the CTI model from [4], and second, we introduce a system for software threat assessment that utilizes CTIO for analyzing Sysmon logs and classifying executed software instances into different threat levels (high, medium, low, and unknown), augmenting defenders cyber defense capabilities through situational awareness, prediction, and automated courses of action.

The rest of the paper is organized as follows. Section II.2 explains the importance of utilizing cyber threat intelligence and engaging in information sharing as part of an organization's security operations and discusses how establishing a robust, structured, and expressive cyber threat intelligence knowledge base can strengthen the security posture. Section II.3 presents CTIO and elaborates on its composition. Section II.4 presents a software threat assessment system that utilizes CTIO and its underlying knowledge base to classify software instances into different threat levels based on the analysis of continuous Sysmon log streams. Section II.5 discusses considerations regarding the presented approach. Section II.6 concludes the paper.

II.2 Threat Intelligence

Threat intelligence can be described as the aggregation, transformation, analysis, interpretation, and enrichment of threat information to provide the necessary context needed for decision-making [3]. Threat information is any information that can help an organization protect itself against a threat. In a blog post[2], Ryan Stillions emphasized that security teams of low threat detection maturity and skills would be able to detect attacks in terms of low-level technical observations without necessarily understanding their significance. On the other hand, security teams of high detection maturity and skills are assumed to be able to interpret technical observations in the sense that the type of attack, the attack methods used, the goals, and possibly the identity of the attacker can be determined.

Threat intelligence sharing allows one organization's detection to become another's prevention by leveraging collective knowledge, experiences, and capabilities to understand better the threats an organization might face. Benefits of threat intelligence sharing include greater insight into cyber threats and

[1] https://github.com/MHaggis/sysmon-dfir
[2] http://ryanstillions.blogspot.no/2014/04/the-dml-model_21.html

enhanced detective and preventive capabilities of an entire community at the strategic, operational, tactical, and technical levels [1]. Machine-to-machine threat intelligence sharing is facilitated by utilizing machine-readable sharing standards that feed relevant, accurate, timely, and actionable intelligence to threat intelligence platforms. An example is the Structured Threat Information eXpression (STIX) language which is currently the most used standard for sharing structured threat intelligence [7].

II.2.1 A Knowledge Base of Threat Intelligence

A knowledge base is a repository of complex structured and unstructured information that represents facts about the world. A knowledge base can evolve over time and utilize codified logic to infer new facts or highlight inconsistencies. Ontology is a form of knowledge representation that defines semantic concepts and their relationships to elucidate a domain of interest. The agreed-upon schema and unambiguous concepts of an ontology allow information to be structured and form a knowledge base that is queryable and can support reasoning using formal logic.

In previous work [4], we argued that a comprehensive ontology for cyber threat intelligence would allow organizations of any size to improve their threat detection, prioritization, and response capabilities. Following up, in this work, we developed the Cyber Threat Intelligence Ontology (CTIO).

II.3 Cyber Threat Intelligence Ontology

Part of our work was to develop a comprehensive Cyber Threat Intelligence Ontology. To achieve that and for supporting interoperability and making the ingestion of cyber threat intelligence into CTIO the least cumbersome, we mainly utilized and interpreted existing works like cyber threat intelligence relevant taxonomies, vocabularies, knowledge bases, and ontologies widely used by defenders. CTIO represents different information types ranging from low-level technical observables to high-level behavioral characteristics, like facts about threat actors, their motivations, their goals and strategies, specific attack patterns and procedures (TTPs), malware, general tools and infrastructures used in adversarial attacks, indicators of compromise, atomic indicators, targets, software weaknesses and vulnerabilities, and courses of action.

We used the web ontology language (OWL) and followed an agile approach for developing the ontology. CTIO comprises several interconnected sub-ontologies based on existing universally utilized taxonomies, such as the Common Vulnerabilities and Exposures (CVE), National Vulnerability Database (NVD), Common Vulnerability Scoring System (CVSS 2.0), Common Platform Enumeration (CPE), Common Weakness Enumeration (CWE), Common Attack Patterns Enumerations and Characteristics (CAPEC), Threat Agent Library (TAL), Threat Agent Motivation (TAM), Adversarial Tactics, Techniques and Common Knowledge (ATT&CK), sharing standards like STIX 2.1 and OpenIOC,

and domain expertise that allowed us to develop a malware ontology and extend the existing CPE schema (ExtendedCPE) to make it more expressive based on our needs. Figure II.1 illustrates the interrelationships between the aforementioned concepts. For further information regarding the taxonomies and sharing standards mentioned above and how they relate to the Cyber Threat Intelligence model, refer to [4].

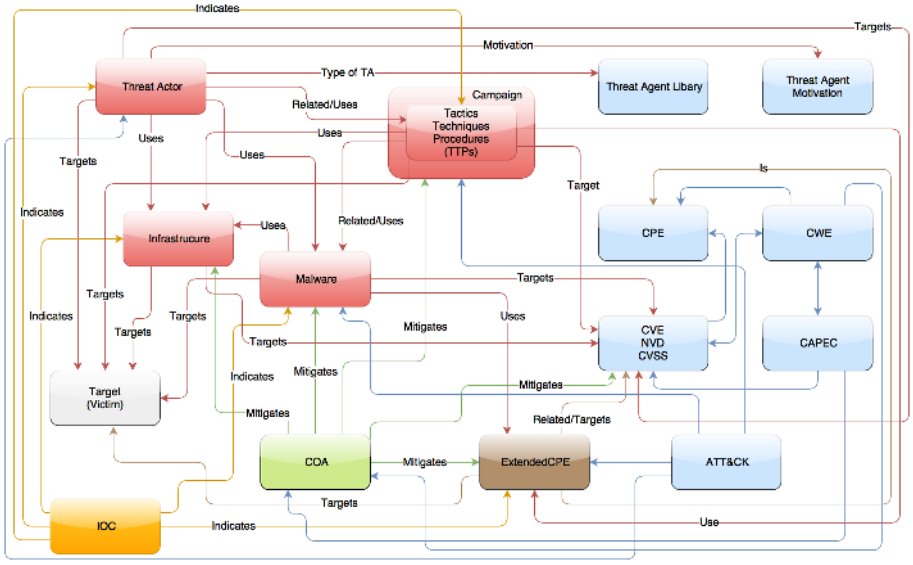

Figure II.1: High-Level Concepts and Relationships of the Cyber Threat Intelligence Ontology

The malware and the ExtendedCPE ontologies are the two major components highly queried in the threat assessment system described in section II.4, and they are intended to represent accurate knowledge of malicious and non-malicious software. All the aforementioned ontologies compose a larger unified ontology for representing comprehensive cyber threat intelligence. OWL constructs are used to perform inference and consistency checking over the knowledge base. For example, to classify software as ExtendedCPE, which is a form of whitelist, requires all the classification criteria of CPE to be met and, additionally, to include a process hash followed up by a programmatic verification function confirming that the software classified is deemed non-malicious. It should be mentioned that ExtendedCPE aims to aggregate non-malicious software but includes software with known or unknown vulnerabilities or benign software that has been utilized in adversarial attacks (e.g., command line tools, browsers, vulnerability scanners, network scanners); hence software within the ExtendedCPE subontology can be associated with different threat levels.

The malware ontology was initially developed based on the STIX 2.1 malware

object and was later enriched with several other properties to assist our automated software assessment methodology. For example, we included properties that increase the possibility of detecting malware based on Sysmon logs' information, such as hashes and dynamic-link libraries that were loaded during the execution of a malware.

The modularity of CTIO allows utilizing existing ontologies and introducing additional concepts into the main ontology skeleton with minimal integration complexity. Information and documentation about CTIO can be found on GitHub[3].

II.4 Software Threat Assessment System

The second contribution of this research work is a system (Figure II.2) that utilizes Sysmon logs, cyber threat intelligence, and formal logic to classify executed software on endpoints as of high, medium, low, or unknown threat level based on technical or behavioral characteristics defined in a policy (Table II.1) and encoded into the ontology. Thus, organizations can increase their threat awareness capability and partly automate a process for detecting malicious or suspicious software instances on their infrastructure. Also, cyber threat intelligence allows defenders to better understand the threat and how to respond.

The system handles available threat intelligence multi-purposely. Not only can it identify malware based on a principled and systematic analysis but can improve the overall cyber defense operations through increased situational awareness, prediction, and descriptive or machine-executable courses of action.

Threat Level	Characteristics
High	Malicious software
	Benign software but with relationship to malicious indicator(s)
	Unknown software but with relationship to malicious indicator(s)
Medium	Benign software but vulnerable
	Benign software but has been used by threat actor to perform attack
Low	Possibly non-malicious software
Unknown	Unknown software without known relationship to malicious indicator(s)

Table II.1: Example Threat Level Classification Policy

Situational awareness: is achieved through the evidence-based knowledge accumulated within the ontology. A simple observable such as an IP, domain name, hash, or registry key can be part of or related to an indicator of compromise captured within the knowledge base and be queried upon to retrieve more contextual information based on what is known. For example, an identified malicious hash can be pivoted to provide related information about command and

[3]https://github.com/Vasileios-Mavroeidis/CTIO

control (C2) servers that this malware instance has been observed communicating with, the malware family that belongs to, the campaigns that have utilized this malware instance or another instance of the same family, the threat actor behind the identified campaign and malware, the motivations and goals of the threat actor, as well as the target of the attack such as a specific sector the malware family and the attacker target. When an incident's scope can be determined and taken into account, the response speed and effectiveness increase.

Prediction: an organization can introduce an anticipatory threat reduction element into security operations through the increased levels of situational awareness attained from utilizing cyber threat intelligence. For instance, at a more technical level, an unknown executed software that relates to a known malicious property may support revealing an associated malware family. A defender can potentially infer the subsequent steps of a campaign targeting the organization or quickly get an insight into what the attack possibly has caused.

Course of action: refers to the steps taken either to prevent an attack or respond to an attack. A course of action within CTIO is described in prose or in a standardized manner that enables real-time automated response actions. Our system utilizes the OASIS Open Command and Control (OpenC2) language [5]. OpenC2 enables the command and control of cyber defense systems and components in a manner that is agnostic of the underlying utilized products, technologies, transport mechanisms, or other aspects of the implementation. An OpenC2 command comprises an action, a target, an optional actuator that executes the command, and additional arguments that influence how the command is performed. OpenC2 assumes that an event has been detected, a decision to act has been made, the action is warranted, and the initiator and recipient of the commands are authenticated and authorized [6].

Other advantages of the proposed system are the following:

- Integrating and updating new and existing concepts and threat intelligence is achieved seamlessly or requires minimal modifications due to the system's underlying ontology language technology. Therefore CTIO can be enriched structurally and updated about emerging threats.

- Sysmon log analysis can help detect threats that could otherwise go undetected by traditional network intrusion detection systems and network firewalls, such as encrypted traffic.

- The inference capability of ontologies by using logic, the available constructs, and class expressions can derive very expressive knowledge representations increasing data unification and interpretability. For example, a set of rules can classify new malware instances based on the infrastructure type they use, like malware that has used cloud service APIs to exfiltrate data or malware related to establish a botnet and botnet infrastructure. Also, consistency checking is vital to avoid misrepresentation of data.

- The ontological knowledge base can be searched using granular semantic queries allowing human or machine agents to answer complex questions or to perform threat hunting. Queries can also be enriched with regular expressions formulating a more signature-based detection method.
- The proposed system can speed up security operations, improve the detection rate of non-benign software, and add an additional layer of security by automating the investigation process.
- The cyber threat intelligence ontology can scale, be deployed in a cloud environment, and be maintained by an organization or a threat intelligence community. In our case, CTIO is accessed using rest-style SPARQL queries over HTTPS.

II.4.1 Operational Flow of the System

The system, also presented in Figure II.2, *aggregates* Sysmon logs from Windows-based workstations and, using a *parsing engine*, automatically extracts attributes based on each log's Event ID for conducting a threat assessment. For example, a log with Event ID 1 provides detailed information about process creation. Figure II.3 presents a simplified Sysmon log with Event ID 1 linked to the WannaCry ransomware attack manifested in May 2017. The parsing engine extracts multiple elements like Event ID, computer name, username, timestamp, process hash, and command lines of both current and parent processes.

Next, a *lookup engine* inspects whether each process is included in an in-house hash whitelist part of the ExtendedCPE component and retrieves the associated threat level. The threat level of a benign process may change based on new information, such as in the case of a new CVE. Also, benign software instances associated with a particular threat level may be further inspected regarding their behavior. For example, a PowerShell instance spawned by a graphical word processing program will raise a case. Further, a downloaded file (Sysmon Event ID 11) by a PowerShell instance spawned by a graphical word processing program will classify the file as a high threat. Such criteria are encapsulated within ontology expressions allowing an inference engine to deduct new information. Also, the relevant Sysmon events that are to be further investigated are mapped, translated to triples, and are included in a dedicated knowledge base. The *lookup engine* inspects whether other extracted element values such as hashes and command lines have been previously queried within a specified time-period and retrieves the relevant information. This tier retains the processing cycles of the *SPARQL engine* low and verifies rapidly benign or malicious software. In the sight of an already classified process, the system pushes the information directly to the *decision-making process engine*, and the appropriate *course of action* is applied or recommended. Element values of unidentified processes become part of SPARQL queries that perform semantic searches upon the CTIO knowledge base and are further transformed into triples for performing reasoning. Based on the derived information and the codified

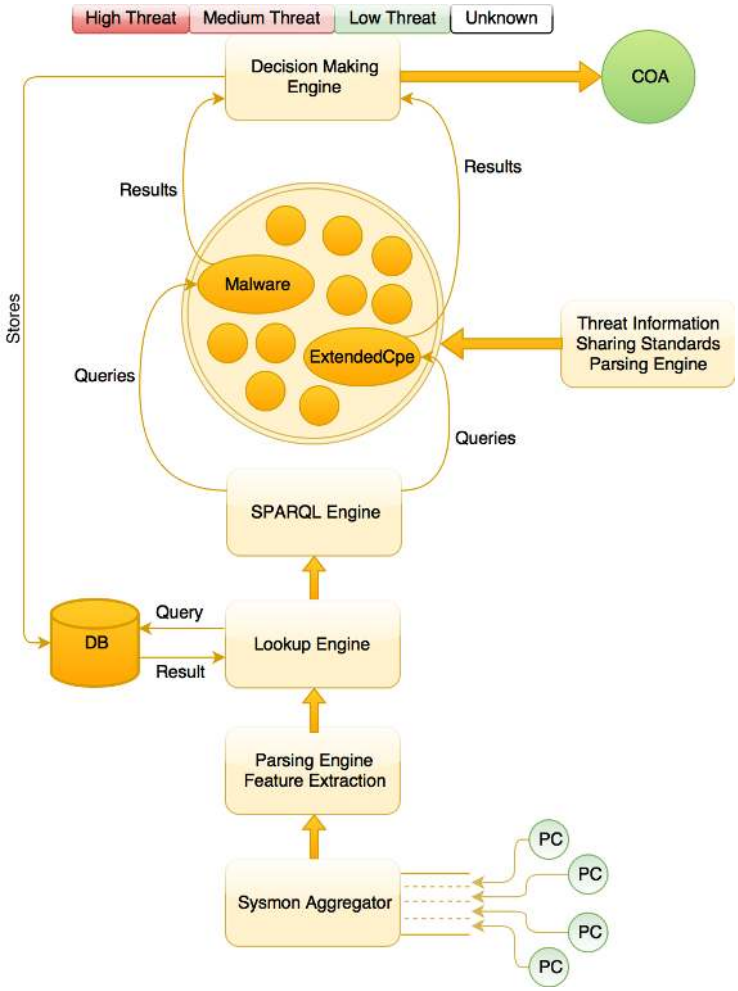

Figure II.2: High-Level Architecture of the Software Threat Assessment System

threat classification rules like the ones presented in Table II.1, the *decision-making process engine* classifies a process as high, medium, low, or unknown threat level. Processes that have been classified unknown are either considered benign after manual verification or are further investigated in timed intervals by being correlated with new intelligence. Furthermore, the system recommends or executes *courses of action* by referencing a course of action type policy and presents relevant threat intelligence to increase threat awareness.

Figure II.4 presents a set of sequential semantic queries based on the WanaCry ransomware process creation Sysmon log presented in Figure II.3.

Given a hash, the first query investigates whether an associated indicator

```
<EventID>1</EventID>
<EventRecordID>3421</EventRecordID>
<TimeCreated SystemTime="2017-10-19T09:03:15.909515800Z"/>
<Execution ProcessID="7792" ThreadID="7836"/>
<Computer>DESKTOP-K9E7OJ2</Computer>
<Security UserID="S-1-5-18"/>
<EventData>
<Data Name="UtcTime">2017-10-19 09:03:15.878</Data>
<Data Name="ProcessGuid">{19048A7C-6A53-59E8-0000-0010709AA501}</Data>
<Data Name="ProcessId">1776</Data>
<Data Name="Image">C:\Users\win10\AppData\Local\Temp\Temp1_Ransomware.WannaCry.zip\tasksche.exe</Data>
<Data Name="CommandLine">"C:\Users\win10\AppData\Local\Temp\Temp1_Ransomware.WannaCry.zip\tasksche.exe"</Data>
<Data Name="CurrentDirectory">C:\Windows\system32\</Data>
<Data Name="User">DESKTOP-K9E7OJ2\win10</Data>
<Data Name="LogonGuid">{19048A7C-393E-59E7-0000-0020A6430300}</Data>
<Data Name="LogonId">0x343a6</Data>
<Data Name="TerminalSessionId">1</Data>
<Data Name="IntegrityLevel">Medium</Data>
<Data Name="Hashes">MD5=84C82835A5D21BBCF75A61706D8AB549,
                    SHA256=ED01EBFBC9EB5BBEA545AF4D01BF5F1071661840480439C6E5BABE8E080E41AA,
                    IMPHASH=68f013d7437aa653a8a98a05807afeb1</Data>
<Data Name="ParentProcessGuid">{19048A7C-393F-59E7-0000-00107E960300}</Data>
<Data Name="ParentProcessId">3384</Data>
<Data Name="ParentImage">C:\Windows\explorer.exe</Data>
<Data Name="ParentCommandLine">C:\Windows\Explorer.EXE</Data>
</EventData>
```

Figure II.3: Event ID 1 Sysmon Log Related to WannaCry Ransomware

```
ASK { ?malware mwr:hasIOC mwr:ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439ce5babe8e080e41aa }
```

```
SELECT DISTINCT ?openc2
WHERE { ?malware mwr:hasIOC mwr:ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.
        OPTIONAL { ?malware mwr:hasOpenC2 ?coa.
                   ?coa coa:openC2 ?openc2 } }
```

```
CONSTRUCT { { mwr:b80d53158dc3bd84f7749b4c54780cf6f82aa3993507b26c8f733173a729caf2 ?predicate ?object } }
```

Figure II.4: SPARQL Queries

of compromise exists in the knowledge base. Having confirmed an existing indicator of compromise, the system based on the codified inference statements defined in the associated policy has inferred that the process is of high threat. The second query requests a *course of action* to implement and is forwarded to the *decision-making process engine* that, based on a course of action type policy, allows or disallows execution. For instance, in the case of WannaCry, a course of action constitutes allowing traffic passing through a firewall for a specific domain that acts as a kill-switch, blocking C2 communications to specific .onion domains, for externally facing servers and systems that do not use SMB or Windows Network File Sharing capabilities block SMB network traffic, and finally restore infected systems to a previous state. Examples of OpenC2 commands are presented in Figure II.5.

```
{
    "action":"allow",
    "target": {
        "network_traffic": {
            "domain_name": {
                "value":"iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea"
            }
        }
    },
    "actuator": {
        "application_level_proxy": {}
    }
}
```

(a)

```
{
    "action":"deny",
    "target": {
        "network_traffic": {
            "domain_name": {
                "value":"57g7spgrzlojinas.onion",
                "value":"76jdd2ir2embyv47.onion",
                "value":"cwwnhwhlz52maqm7.onion",
                "value":"gx7ekbenv2riucmf.onion",
                "value":"xxlvbrloxvriy2c5.onion"
            }
        }
    },
    "actuator": {
        "application_level_proxy": {}
    }
}
```

(b)

```
{
    "action":"deny",
    "target": {
        "network_traffic": {
            "port": {
                "number":"137",
                "number":"138",
                "number":"139",
                "number":"445",
            }
        }
    },
    "actuator": {
        "network_firewall": {}
    }
}
```

(c)

```
{
    "action":"restore",
    "target": {
        "device": {
            "computer": {
                "name":"DESKTOP-K9E7OJ2"
            }
        },
        "options":{
            "restore_point":"dateTime"
        }
    },
    "actuator": {
        "WindowsServer": {}
    }
}
```

(d)

Figure II.5: OpenC2 Course of Action for WannaCry Ransomware

Additionally, the system returns a set of RDF triples that comprise the complete known to our organization knowledge regarding the identified threat (third query). Figure II.6 presents a high-level threat intelligence graph of the referenced WannaCry ransomware.

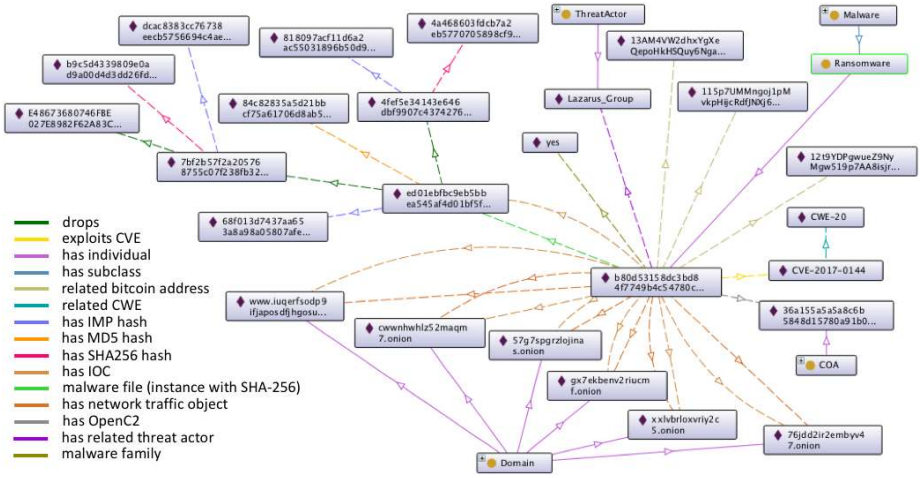

Figure II.6: RDF Graph of WannaCry Ransomware

II.5 Discussion

This research work presented an approach for software threat assessment that relies on Sysmon logs and a cyber threat intelligence ontology to evaluate and infer the threat level of instantiated software in a system automatically. The general system architecture and the underlying ontology are not restrictive to utilizing Sysmon logs but, in the same way, can utilize a diversified set of log types.

The proposed approach elucidated the benefits derived from utilizing ontology technology and logic for cyber threat intelligence purposes, where manual-based approaches often hinder the complex tasks of correlation, analysis, and inference. An ontology for cyber threat intelligence comprises multiple concepts describing the who, what, why, when, where, and how of adversarial operations and can be integrated into many different functions of cyber defense such as risk management, threat hunting, incident response, or proactive defense.

Performing core reasoning tasks and semantic queries on large and complex ontologies are resource and time-intensive. Scaling such ontological systems should be considered by taking into account the size of the knowledge base, the number and complexity of the expressions and rules applied, and the frequency for applying reasoning on new intelligence.

Finally, elevating the standard RDF tabular representation to visualized semantic graphs provides better and easier knowledge exploration and, consequently, conveys key insights more effectively.

II.6 Conclusion

Defenders utilize cyber threat intelligence to make threat-informed decisions. In this research work, we presented a semantic representation of a cyber threat intelligence model [4] using the web ontology language for the purpose of introducing automation in assessing the threat level of instances of executed software on endpoints. Using the reasoning capability of ontologies, we codified statements that can infer the threat level of a process by correlating information derived from Sysmon logs to cyber threat intelligence. In addition, we demonstrated how a standardized language for command and control could activate a rapid threat-informed response.

Acknowledgements. This research was supported by the research project Oslo Analytics (Grant No. 247648) funded by the Research Council of Norway.

References

[1] Chismon, D. and Ruks, M. *Threat Intelligence: Collecting, Analysing, Evaluating*. Tech. rep. MWR Infosecurity, UK Cert, United Kingdom, 2015.

[2] Iannacone, M. et al. "Developing an Ontology for Cyber Security Knowledge Graphs". In: *Proceedings of the 10th Annual Cyber and Information Security Research Conference*. ACM. 2015.

[3] Johnson, C. et al. *Guide to Cyber Threat Information Sharing*. NIST Special Publication. NIST, 2016.

[4] Mavroeidis, V. and Bromander, S. "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence". In: *Proceedings of the 2017 European Intelligence and Security Informatics Conference (EISIC)*. IEEE. 2017, pp. 91–98.

[5] Mavroeidis, V. and Brule, J. "A Nonproprietary Language for the Command and Control of Cyber Defenses – OpenC2". In: *Computers & Security* vol. 97 (2020).

[6] OASIS. *Open Command and Control (OpenC2)*. Accessed: 2017. [Online]. Available: https://www.oasis-open.org/committees/openc2/.

[7] Sauerwein, C. et al. "Threat Intelligence Sharing Platforms: An Exploratory Study of Software Vendors and Research Perspectives". In: *Proceedings of the 13th Internationale Tagung Wirtschaftsinformatik (WI)*. Ed. by Leimeister, J. M. and Brenner, W. 2017.

[8] Syed, Z. et al. “UCO: A Unified Cybersecurity Ontology”. In: *Proceedings of the AAAI Workshop on Artificial Intelligence for Cyber Security*. AAAI Press. 2016.

[9] *Unified Cyber Ontology*. [Online]. Available: https://github.com/ucoProject/uco. GitHub Repository. 2016.

Paper III

Threat Actor Type Inference and Characterization within Cyber Threat Intelligence

Vasileios Mavroeidis[1], Ryan Hohimer[2], Tim Casey[3], Audun Jøsang[4]

Published in: *Proceedings of the 13th International Conference on Cyber Conflict (CyCon 2021)*, May 2021, pp. 327–352, NATO CCDCOE Publications/IEEE.

Abstract

As the cyber threat landscape is constantly becoming increasingly complex and polymorphic, the more critical it becomes to understand the enemy and its modus operandi for anticipatory threat reduction. Even though the cyber security community has developed a certain maturity in describing and sharing technical indicators for informing defense components, we still struggle with non-uniform, unstructured, and ambiguous higher-level information, such as the threat actor context, thereby limiting our ability to correlate with different sources to derive more contextual, accurate, and relevant intelligence. We see the need to overcome this limitation in order to increase our ability to produce and better operationalize cyber threat intelligence. Our research demonstrates how commonly agreed upon controlled vocabularies for characterizing threat actors and their operations can be used to enrich cyber threat intelligence and infer new information at a higher contextual level that is explicable and queryable. In particular, we present an ontological approach to automatically inferring the types of threat actors based on their personas, understanding their nature, and capturing polymorphism and changes in their behavior and characteristics over time. Such an approach not only enables interoperability by providing a structured way and means for sharing highly contextual cyber threat intelligence but also derives new information at machine speed and minimizes cognitive biases that manual classification approaches entail.

[1]University of Oslo, Oslo, Norway, vasileim@ifi.uio.no
[2]DarkLight Inc., Richland, Washington, United States, ryan.hohimer@darklight.ai
[3]Intel Corp., Chandler, Arizona, United States, tim.casey@intel.com
[4]University of Oslo, Oslo, Norway, audun.josang@mn.uio.no

III.1 Introduction

Cyber threat intelligence (CTI) is undeniably an essential element for building a robust security posture against adversarial attacks. Establishing a threat intelligence program allows security teams to benefit from increased situational awareness, and thus minimize their organizations' attack surfaces. Evidence-based knowledge of both adversary dynamics and an organization's attack surface can support anticipatory threat reduction. Organizations follow a process of increasing maturity with respect to their cyber capability, transitioning from manual and reactive approaches to more automated and proactive.

Proactive cyber defense is intelligence-driven and focuses on providing awareness and preparing an organization against anticipated attacks. Every adversarial attack can be decomposed into elements that provide information about the *who*, *what*, *where*, *when*, *why*, and *how*. The *who*, commonly known as attribution, identifies the individual, group, organization, or nation that conducted the adversarial operation. The *what* reflects the scope of the attack. The *where* relates to the attack's direction, such as where it is coming from and its target – an organization, industry, or country. The *when* can be perceived as the timestamp of the attack and can be deterministic or probabilistic. The *why* is equivalent to motivation and designates the goals and the objectives of the adversary. The *how* is made up of the tactics, techniques, and procedures (TTPs) employed by the adversary for conducting the operation. Collectively, these factors provide insight into how adversaries plan, conduct, and sustain their operations.

Attribution is typically a challenging task requiring direct evidence through a principled and systematic analysis which correlates multiple internal and external data sources and threat intelligence. Such a process identifies and maps TTPs and associated tools and infrastructure to known sources of similar attacks. However, threat actors intend to remain unidentified and employ deception and obfuscation techniques that can lead to incorrect attribution or weakening the possibility of correctly associating a particular activity with a known adversary. For example, the Russia-backed group Turla (also known as Waterbug) was discovered to be using the infrastructure and malware of APT34 (also known as OilRig), an Iranian threat group [18]. Nevertheless, many times, a threat actor profile is created and linked to one or more adversarial operations based on common identifiable properties without actual attribution, meaning that the adversary's real-world identity remains unknown.

Capturing high-level information such as the motives behind an adversarial operation and contextualizing technical findings; for example, by estimating the level of sophistication, skills, and resources needed to plan and execute the attack, can characterize the perpetrator and infer its nature even when direct attribution has not been achieved. The opposite is also plausible. The nature of a perpetrator reflects its capability, persistence, and motives. In addition, in a threat landscape that has become very diversified and hybridized, the importance of portraying adversaries and their nature as threat actor types is apparent. Threat actors are continuously evolving and are becoming polymorphic with

multiple motivations and goals. Existing approaches in characterizing threat actors and their operations mostly fall under the category of regular intelligence reports that fail to capture information in a specific representation format that both humans and machines can interpret. On the other side lies purely technical information intended to be consumed directly by cyber defense products.

A wide range of threat actor types exists, ranging from disgruntled employees to organized cyber crime and nation-state-backed groups. Threat actors have specific traits common to most of their behaviors. For example, an employee with a grudge against their organization is motivated by disgruntlement. In contrast, a state-sponsored group may aim to achieve dominance over another nation for geopolitical reasons. To operationalize this type of characterization, we need to satisfy two criteria. First, the definitions of actor types must be unambiguous, and second, we must characterize them using a set of attributes that enables robust, reliable enumeration and inference.

This research reflects the operational and strategic benefits derived from semantically portraying threat actors as threat actor types (e.g., nation-state, hacktivist, terrorist, organized cyber crime) to understand the actors' nature and capture polymorphism and changes in their behavior and characteristics over time. Furthermore, we present an ontological system for threat actor type inference which relies on a standard set of attributes for characterizing threat actors and their operations. Axioms (expressions) capture domain knowledge regarding the composition of threat actor types based on their defining attributes. The presented approach can augment existing static enumerative approaches for threat actor type classification with a flexible generative system based on the logic encapsulated in the ontologies. Such an approach enables machine understanding and logical reasoning based on that understanding with transparent and explicable results. The proof-of-concept ontology we engineered utilizes Casey's Threat Agent Library (TAL) [4]. The original TAL typology has been refined and can be updated further to reflect a more contemporary description of threat actor types and their defining attributes.

A semantically expressed threat actor typology based on a set of standard characterization attributes offers the following advantages.

- Based on commonly agreed upon definitions, a machine-understandable interpretation of threat actor types and their defining attributes eliminates ambiguity regarding their meaning by annotating their unique characteristics. The term *commonly* above refers to the need for interoperability. A standard vocabulary and representation for threat actor types can be integrated across different technologies such as threat intelligence platforms and threat information sharing languages, and used when generating threat reports. For example, people often interpret seemingly simple terms such as hacktivist differently. Correlating a threat actor type with an operation is then subject to fallacies when the semantics for what comprises a particular type are not in place. This makes shareable information inaccurate and

contradictory since different entities may have different interpretations of the same term, leading to inconsistent threat actor profiles. In this research, each threat actor type is semantically bound to a specific set of attribute values, thereby making it unique by providing context as to what comprises a particular type.

- Representing domain knowledge in a declarative form such as axioms and facts can enable automatic inference via the ability of machines to reach a conclusion based on evidence. In this research, axioms capture the unique attribute combinations that characterize different threat actor types. Using a description logics reasoner, also known as an inference engine, instances of threat actors can be programmatically examined to infer their type. Automatic inference also speeds up traditional analytical processes that require testing competing hypotheses about the adversary's type to be tested.

- Polymorphism and changes in threat actor behavior over time are becoming common, with adversaries being influenced by different motives and goals. Some threat actors evolve in nature and gradually engage in larger-scale and more complex operations. In contrast, others pause their operations, disappear, or even go through organizational changes like establishing new units. It is essential for the threat intelligence community to recognize and formally represent polymorphism and behavioral changes over time so that threat actor profiles can evidentially account for more than one threat actor type (Figure III.1). For example, as presented in Section III.5, the state-sponsored Lazarus Group has engaged in activities not only motivated by geopolitical reasons to achieve dominance over other nations by conducting stealthy cyber espionage campaigns but also for nationalistic reasons and revenge by engaging in destructive hacking, as well as for financially motivated reasons by conducting bank heists possibly to fund their operations. As discussed later, available threat actor knowledge bases appear to fail to capture polymorphism and behavioral changes, resulting in monolithic representations that lack evidence-based relationships concerning the derivation of their characterization. In addition, most of the time, the characterizations are based on proprietary works that are also ambiguous due to nonexistent or insufficient definitions. Ambiguity and imprecision create confusion and diminish the value of intelligence in cyber operations.

- The definition and utilization of characterization attributes (e.g., motivations, goals, objectives, visibility) can contextually enrich cyber threat intelligence. Furthermore, a semantic representation of threat actor types based on those attributes enables granular querying of higher contextual precision that can answer complex questions. In proactive cyber defense, we want to answer questions such as: "*Based on the fact that my organization is within the [finance, government, healthcare, etc.] sector and I have knowledge of the assets [infrastructure, software, data, etc.] I own*

and need to protect; I want all relevant information about threat actors and operations that currently target institutions similar to mine within my sector and preferably in the country my organization is located". Precision in querying when using the characterization attributes and the threat actor types can provide more contextual and insightful results. For example, the above question could be refined to: "*Based on the fact that my organization is within the [finance, government, healthcare, etc.] sector and I have knowledge of the assets [infrastructure, software, data, etc.] I own and need to protect; I want all relevant information about threat actors and operations* ***in the current calendar year*** *that target institutions within my sector,* ***in my country, are classified as a nation state, and have also been observed to engage in financially motivated cyber crime***". The derived intelligence can provide defenders with increased situational awareness and thus allow them to better prioritize their defense efforts according to their most relevant threats.

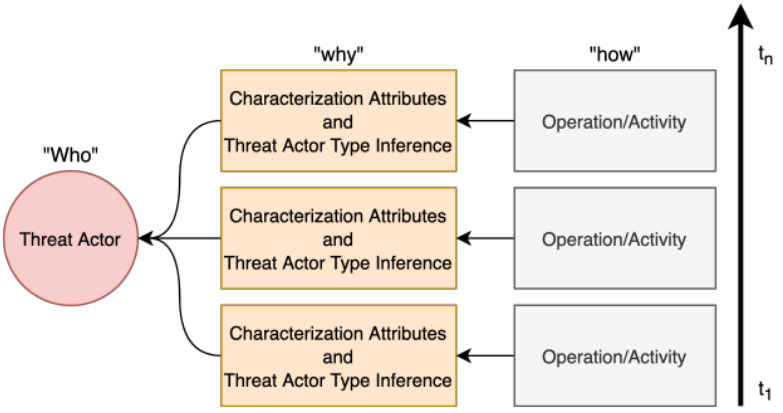

Figure III.1: Semantic Modeling of Threat Actor Characterization

The rest of the paper is organized as follows. Section III.2 presents background information pertinent to cyber threat intelligence, introduces the Threat Agent Library [4] that was referenced to create a prototype ontology for threat actor type inference, and presents and analyzes different threat actor knowledge bases with respect to how they handle high-level contextual information in terms of ambiguity, structured shareability, explainability, and most importantly operationalization ability. Additionally, Section III.2 discusses how the Structured Threat Information eXpression (STIX) language deals with interpreting threat actor polymorphism. Section III.3 discusses knowledge representation and ontology engineering within the cyber threat intelligence domain, and annotates how ontology inference can provide defenders with additional information and insights at machine speed. Section III.4 presents an ontology for threat actor characterization and threat actor type inference.

Section III.5 validates the proposed concept's efficacy and presents a use-case analysis where the ontology presented in Section III.4 is used to infer threat actor types automatically. Furthermore, Section III.5 demonstrates the potential of characterization attributes in providing highly contextual and queryable cyber threat intelligence. Finally, Section III.6 concludes the paper.

III.2 Background Information

III.2.1 Cyber Threat Intelligence

Cyber threat intelligence is actionable information about adversaries and their activities. To be of value, cyber threat intelligence needs to be timely, accurate, and relevant to deliver the essential context needed to support the decision-making processes, prioritize the implementation of controls, and the allocation of often limited defensive resources. Adopting a four-tier model, we have the following types of threat intelligence.

Technical cyber threat intelligence comprises observables and indicators of compromise (IOCs) with additional context associated with known attacks and can be consumed directly by cyber defense components.

Tactical cyber threat intelligence focuses on threat actor TTPs and tools, as well as their methods to avoid detection. Security teams use tactical information to make informed decisions about building a defense strategy to mitigate those attacks. If a defender can detect or prevent attacker behavior compared to simply utilizing basic artifacts like file hashes or IP addresses, they make it more costly and painful for an attacker to pivot their path [2]. A knowledge base with adversarial TTPs is MITRE ATT&CK (Adversarial Tactics, Techniques Common Knowledge). The MITRE ATT&CK Groups knowledge base is discussed in Section III.2.4.

Operational cyber threat intelligence is contextual and provides a detailed insight about the nature, motive, timing, goals, and the mechanics of a particular attack. Operational cyber threat intelligence also includes technical information to provide a more complete and actionable picture of an ongoing incident.

Strategic cyber threat intelligence is nontechnical and demystifies an organization's existing and forecasted threat landscape and drives its high-level strategy. It informs about emerging threats relevant to an organization's profile and considers how prepared an organization is to defend.

III.2.2 Threat Agent Library

Introduced in 2007, the Threat Agent Library (TAL) [4] is a set of definitions and descriptions to represent significant threat agent categories, or as termed in this paper, threat actor types. The TAL was developed to support risk management processes by simplifying the identification of threat agent archetypes that pose the most significant risk to specific assets (Figure III.2). Based on the available information on each archetype class, an organization can get an insight into current adversarial activities and consequently take action to improve its

security posture. The library (Table 1) enumerates twenty-one archetypes (e.g., government spy, radical activist, untrained employee, disgruntled employee) and their associated defining attributes: access, outcome, limits, resources, skills, objective, visibility, and motivation. The defining attributes reflect the typical characteristics of each threat actor type.

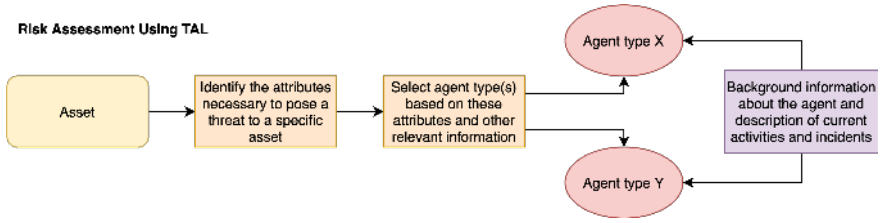

Figure III.2: Risk Assessment Using the Threat Agent Library

This research presents a proof-of-concept ontological representation of TAL, with minor improvements, for automatically inferring threat actor types from cyber threat intelligence instances (objects). The decision to use TAL is based on its assessment of combinations of characterization attributes that uniquely identify different threat actor types. Further, we emphasize the importance of having a set of standard characterization attributes to contextualize cyber threat intelligence, thereby making it more actionable and relevant. We also argue that modeling approaches should be temporal-based to capture threat actor polymorphism and behavioral changes over time. As presented in the next sections, available threat actor knowledge bases struggle to capture such formalisms resulting in contextual loss and ambiguity.

III.2.3 Threat Actor Characterization Using STIX 2.1

Structured Threat Information eXpression (STIX) is a schema that defines a taxonomy for cyber threat intelligence. We discuss and analyze STIX version 2.1 [15] for two reasons. First, because of its ability to describe threat actors, threat actor activity, and their associated characteristics in a machine-readable format, and second, because it has been embraced as the standard representation format for sharing cyber threat intelligence in a structured manner.

The **STIX Threat Actor object** is used for attribution and aggregates information about threat actors, such as their goals, motivations, sophistication, resource-level, and type. Additionally, it utilizes relationship objects to reference objects that represent the actual identity behind a threat actor (be it a human or organization), the tools (e.g., malware) that the actor has been known to use or used in a specific attack, the patterns of attack (e.g., attack patterns) that the actor is known to follow, the location where the actor is believed to be, infrastructure both owned and compromised that the actor is known to use, as well as attributes about the actor that help characterize them. This is an object of high value in proactive cyber defense where strategic, operational, and tactical cyber threat intelligence play a significant role. Figure III.3 presents the

STIX threat actor object with its characterization attributes and relationships with other objects.

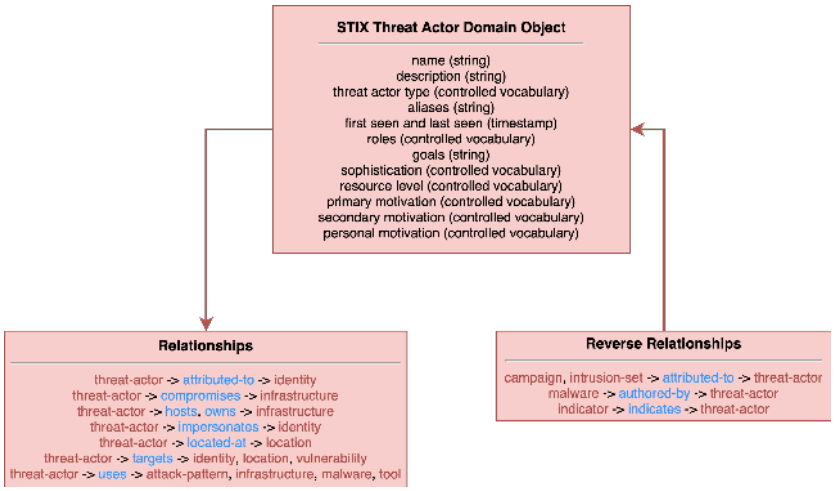

Figure III.3: STIX Threat Actor Object

A critical aspect that the STIX threat actor object does not account for is capturing and semantically representing behavioral polymorphism in a temporal manner, as in the case where a threat actor is conducting different operations than what is known, reflecting a possible change to its primary or secondary motivations and goals. Furthermore, the characterization attributes of the threat actor object do not hold any semantic relationships with other objects to provide direct evidence for justifying the existing characterization. This is especially the case when a threat actor object has more than one value populated for an attribute (e.g., a threat actor that accounts for more than one threat actor type). Also, some of the STIX vocabularies used for characterizing adversaries are ambiguous because they lack definitions. The generation of the threat actor type attribute is a manual and subjective process prone to human fallacies. For example, a threat actor object with the populated threat actor type value "nation-state" and resource-level "individual" (limited resources) is unlikely to be correct but is deemed a valid STIX statement. This reflects the advantage of utilizing an automated generative threat actor type inference approach (see Section III.4) for augmenting existing manual approaches.

Intelligence generation is an evidence-based approach where information should traverse from the more technical and detailed lower strata to higher, more contextualized ones. Such an approach demystifies the misunderstanding of intelligence usage and intelligence generation, meaning that intelligence can be used multi-directionally but is initially created based on a bottom-up approach. For example, the STIX campaign object is crucial for grouping adversarial behaviors that describe a set of malicious activities against a specific set of targets that occur over a period of time. Campaigns can be attributed to threat

actors and can be characterized by their objectives and the incidents they cause, people or resources they target, and the resources (e.g., infrastructure, malware, tools) they use. A campaign object does not hold direct relationships with the characterization attributes of the associated threat actor conducted the campaign but connects to the attributes via the threat actor object, and cannot directly influence the population of those attributes or their updating in a way that can reflect this behavioral change in a temporal manner.

III.2.4 Threat Actor Knowledge Bases

A knowledge base is a collection of information about a particular subject area that can be used to support decision-making and draw conclusions. A knowledge base with information about threat actors' capabilities, goals and motivations, and past and ongoing activities can inform prevention and response strategies. An unstructured knowledge base can be a simple aggregating system such as a collection of threat reports. At a basic level, developing a structured knowledge base requires a schema that defines its structural composition, information sources for populating the knowledge base, and optimally controlled vocabularies for additional context and granular searchability. Describing a threat actor with high confidence demands processing, correlating, analyzing, and integrating different relevant intelligence sources.

This section presents a set of open-source threat actor knowledge bases, and analyzes their structural composition with respect to how easy it is to operationalize them in the context of finding information relevant to our needs.

MITRE ATT&CK [12] is a knowledge base of known adversary tactics and techniques based on openly available analyzed activity. It is a valuable resource to better understand observed adversarial behavior, and it can be used for multiple purposes, such as for adversary emulation, behavioral analytics, cyber threat intelligence enrichment, defense gap assessment, red teaming, and SOC maturity assessment [1]. ATT&CK matrices exist about adversary behavior targeting enterprise environments, mobile, and industrial control systems. Moreover, information pertinent to the software adversaries use, mitigation techniques, procedure examples, and detection recommendations are also available. Further, the associated PRE-ATT&CK matrix focuses on operational techniques known to be utilized before an attacker exploits a particular target network or system.

Of particular importance is the available ATT&CK Groups knowledge base, a list of known adversaries and their associated techniques and software tools. Figure III.4 shows the main components of ATT&CK and their relationships.

One way of getting started with ATT&CK is identifying adversarial groups relevant to an organization, based on whom they have previously targeted, such as similar organizations within the same sector, and then look at their TTPs [14]. TTPs that are commonly used can be prioritized for detection and mitigation. However, the ATT&CK Groups knowledge base lacks proper structurality and relationships between adversaries and their targets and between adversaries and their motivations. Information such as targeted countries and sectors and threat group motivations is embedded within the general description of a group

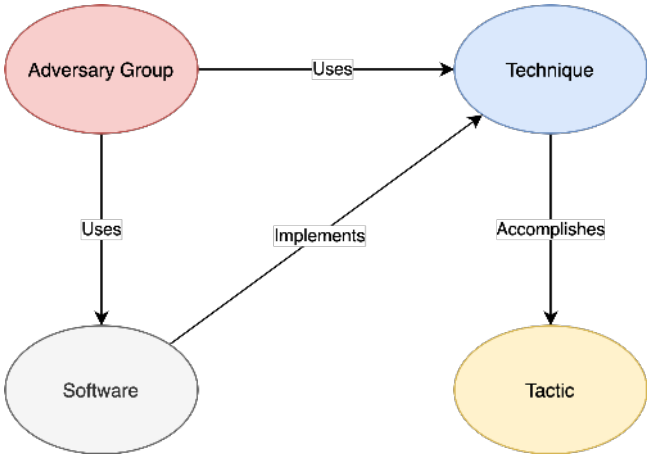

Figure III.4: ATT&CK Model Relationships

and can be unstructurally searched using the ATT&CK portal. However, the vocabularies utilized to specify a group's targets and their motivations are not available, limiting searchability, and consequently, the ability to extract more relevant information. Synergistically, structuring the available information, establishing relationships between concepts, and utilizing a set of standard characterization attributes and other common vocabularies can facilitate more informed and targeted queries over the knowledge base, resulting in getting more relevant, and maybe otherwise missed TTPs to prioritize.

The description of APT19[1] is a good example of unstructured populated information regarding industries the group has targeted.

> *"APT19 is a Chinese-based threat group that has targeted a variety of industries, including defense, finance, energy, pharmaceutical, telecommunications, high tech, education, manufacturing, and legal services. In 2017, a phishing campaign was used to target seven law and investment firms."*

Similarly, the description of APT38[2] is a good example of unstructured populated information regarding a group's motivations.

> *"APT38 is a financially-motivated threat group that is backed by the North Korean regime. The group mainly targets banks and financial institutions and has targeted more than 16 organizations in at least 13 countries since at least 2014."*

The **Threat Actor Encyclopedia** [19] is an effort from Thailand's Computer Emergency Response Team (ThaiCERT) to create a knowledge base

[1] https://attack.mitre.org/groups/G0073/
[2] https://attack.mitre.org/groups/G0082/

of threat group profiles by aggregating, processing, and structuring open-source intelligence. The knowledge base is accessed either by their released document, the ThaiCERT web portal[3], or the available MISP galaxy/cluster that provides a machine-readable representation of the knowledge base (in JavaScript Object Notation) with the ability to be used within the MISP threat intelligence platform. The portal provides a granular search functionality, such as using a source country, a victim country, a victim sector, and motivations as search parameters. As in other efforts, we observed ambiguity and confusion regarding the interpretation and use of characterization attributes. For instance, the threat actor encyclopedia's motivation vocabulary includes the terms *information theft and espionage*, *financial crime*, *financial gain*, and *sabotage and destruction*. Definitions of the above terms have not been provided, making it difficult, for example, to understand the contextual difference between financial gain and financial crime. It can also be argued that *information theft and espionage*, *sabotage and destruction*, and *financial crime* are not motivation types but operation types or intended effects.

The Malware Information Sharing Platform (MISP) is an open-source threat intelligence platform for collecting, storing, and sharing information about cyber security incidents [21]. Due to its open-source nature and modular architecture, the platform can integrate intelligence clusters that, in many cases, are community-driven efforts and can be used to enrich events and attributes.

The **MISP Threat Actor cluster**[4] is a knowledge base of threat groups. The cluster's structural composition is an array of threat group objects that capture information related to the groups, such as name and known aliases, a description, targeted countries and sectors (e.g., private, military, government), their affiliated countries and sponsors, attribution confidence, incident types (e.g., espionage, sabotage or defacement), references relating to the captured knowledge, relations with other groups and operations, and associated malware. A subset of the elements has been derived from the Council on Foreign Relations Cyber Operations[5] vocabulary used for reporting cyber incidents. Like the rest of the knowledge bases investigated, the MISP Threat Actor cluster could benefit from introducing a more expressive structured representation. Currently, multiple characterization attributes are included only in the general description of a threat actor object, making it difficult to parse the information via automated means. For instance, in the example below, the description captures information regarding the motivations, objectives, targeted countries, and the types of operations a group has been observed conducting.

> *"Libyan Scorpions is a malware operation in use since September 2015 and operated by a politically motivated group whose main objective is intelligence gathering, spying on influential and political figures, and operating an espionage campaign within Libya."*

[3]https://apt.thaicert.or.th/cgi-bin/aptgroups.cgi
[4]https://github.com/MISP/misp-galaxy/blob/main/clusters/threat-actor.json
[5]https://www.cfr.org/cyber-operations/

Moreover, the use of different non-standardized vocabularies for enriching the knowledge base and the integration of different intelligence sources for providing additional context introduces ambiguity and confusion. The two shortened examples presented below indicate the importance of utilizing a set of standard characterization attributes with accurate definitions and vocabularies for optimally resolving ambiguity and operationalizing the provided intelligence.

In the example below, *espionage* is used both to describe an incident type and a motive. Additionally, definitions for the available terms are not in place, increasing the probability of misusing the vocabularies.

```
{
  "description": "Anchor Panda is an adversary that CrowdStrike
      has tracked extensively over the last year targeting both
      civilian and military maritime operations...",
  "meta": {
    "attribution-confidence": "50",
    "cfr-suspected-state-sponsor": "China",
    "cfr-suspected-victims": ["United States", "..."],
    "cfr-target-category": ["Government", "..."],
    "cfr-type-of-incident": "Espionage",
    "country": "CN",
    "motive": "Espionage",
    "refs": ["..."],
    "synonyms": ["APT14"]
  },
  "value": "Anchor Panda"
}
```

In the example below, the motive of the group is defined as *Hacktivists-Nationalists* which is reminiscent of a threat actor/group type rather than a motive that influences the actions of an actor.

```
{
  "description": "Turkish nationalist hacktivist group that has
      been active for roughly one year...The group carries out
     distributed denial-of-service (DDoS) attacks and
     defacements against the sites of news organizations and
     governments perceived to be critical of Turkey's policies
     or leadership, and purports to act in defense of Islam",
  "meta": {
    "attribution-confidence": "50",
    "country": "TR",
    "motive": "Hacktivists-Nationalists",
    "synonyms": ["Lion Soldiers Team", "..."]
  },
  "value": "Aslan Neferler Tim"
}
```

III.3 Knowledge Representation and Ontology

Knowledge representation conceptualizes an understanding of the world. It can provide a view of a particular domain of interest and capture that knowledge in a formal representation so that a computer system can utilize it to solve complex tasks, such as inferring new critical information. An ontology is a formalism of knowledge representation that encodes knowledge about a particular domain. An ontology is machine-understandable, holds formal semantics that carry meaning, and allows for reasoning. Formal semantics and logic ensure that the meaning of a concept is unambiguous. An ontology is defined using a knowledge representation language, such as the Web Ontology Language (OWL). An OWL ontology consists of the following three syntactic categories [13]: a sequence of logical *axioms* (statements) that are asserted to be true in the domain being described, *expressions* that represent complex notions in the domain being described (e.g., a class expression describes a set of individuals in terms of the restrictions on the individuals' characteristics), and *entities* such as classes, properties, and individuals, that constitute the basic elements of an ontology. A class represents a concept and provides the means for grouping resources with similar characteristics. For instance, a *threat actor* class can group all known adversaries. Subclasses represent concepts that are more specific than a superclass. For instance, the class *threat actor* can decompose into subclasses that capture a threat actor's intent, such as *hostile* or *nonhostile*, and again decompose into subclasses that define hostile or nonhostile types, such as *nation-state*, *civil activist*, and *untrained employee*. Taking the Lazarus Group as an example and based on available information, it can be classified as a *nation-state* adversary, a subclass of the *hostile* class. The *hostile* class is a subclass of the *threat actor type* class, indicating that the nation-state-backed group Lazarus is an instance of a hostile threat actor. The functional syntax of this example is shown below, with Figure III.5 providing an illustration.

```
Declaration ( Class( :ThreatActorType ) )
Declaration ( Class( :Hostile ) )
Declaration ( Class( :NonHostile ) )
Declaration ( Class( :NationState ) )
Declaration ( Class( :UntrainedEmployee ) )
SubClassOf ( :Hostile :ThreatActorType )
SubClassOf ( :NationState :Hostile )
SubClassOf ( :NonHostile :ThreatActorType )
SubClassOf ( :UntrainedEmployee :NonHostile )
Declaration ( NamedIndividual( :LazarusGroup ) )
ClassAssertion ( :NationState :LazarusGroup )
```

Properties define relationships between individuals (object properties) or between individuals and data type literals (data type properties). For instance, as described in the provided example in Section III.2.4, APT38 is a financially motivated threat group that is backed by the North Korean regime. In addition, APT38 is also known as Stardust Chollima by Crowdstrike [11] and as BlueNoroff

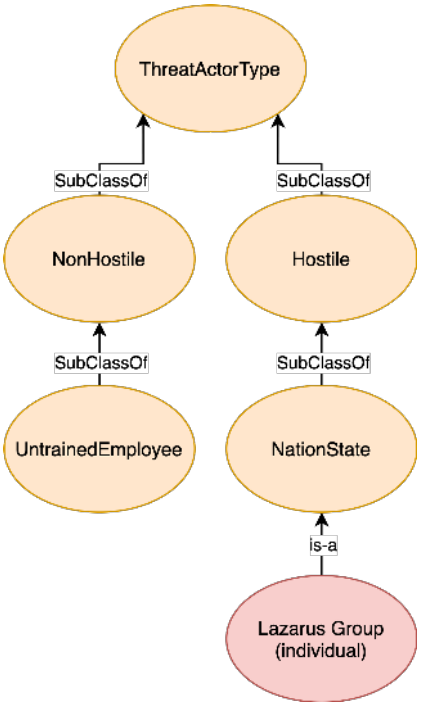

Figure III.5: Example Illustration of Ontology Classes and Subclasses

by Kaspersky [8]. The relation of APT38 with a particular defining motivation and other aliases can be captured by creating relevant object properties and formulating semantic triples. A triple is a set of three entities that codify a statement in the form of subject-predicate-object. This principle is illustrated in Figure III.6, where the arcs represent relations (object properties – predicates), and the ellipticals represent individuals.

OWL offers expressive constructs for reasoning based on description logics. For example, the defined object property, *known-as*, is bidirectional when declared symmetric and allows traversing information when declared transitive. Property declarations can compensate for missing arcs in a knowledge base. A reasoner can parse the knowledge base and infer new information. In the example illustrated in Figure III.6, the symmetric property *known-as* allows inferring that APT38 is known as BlueNoroff and the opposite, such as that BlueNoroff is known as APT38. Furthermore, because of transitivity, a reasoner infers that StarDust Chollima is also known as APT38 (dashed arc) even though it was not directly defined. Ontological axioms, expressions, and constructs can infer information based on causal relationships. For instance, a reasoner will not infer that a threat actor is of *nation-state* type when the resource level property is not populated with the value *government*, according to the class expression that encodes what a nation-state threat actor comprises.

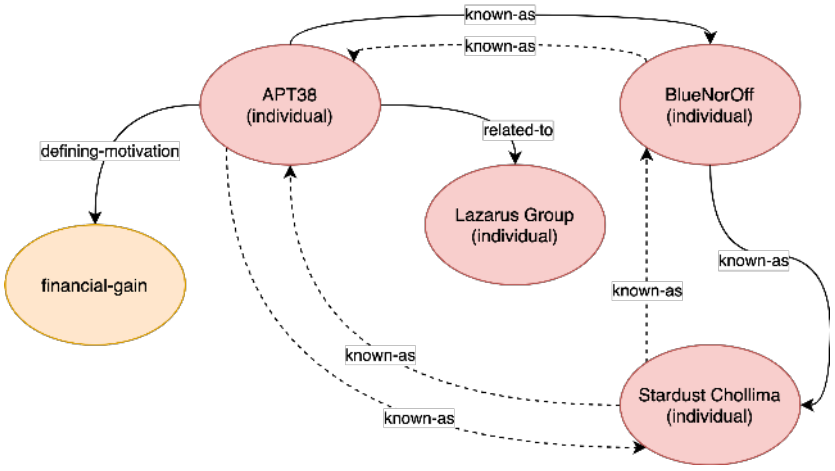

Figure III.6: Semantic Representation of APT38

III.4 A Domain Ontology for Threat Actor Profiling

This section presents a domain ontology for threat actor profiling and actor type inference based on the Threat Agent Library (TAL) [4]. TAL defines threat actor type attributes through controlled vocabularies, such as motivation, access, outcome, limits, resources, skills, objectives, and visibility, and when used collectively, these identify the unique characteristics of each threat actor type. Threat actor types refer to categories that adversaries can be classified into, such as spy, civil activist, and nation-state. In TAL, *threat agent* denotes a class of threat actors and is synonymous with *threat actor type*. The definitions of the TAL terms can be found in [4] and [3].

To develop the ontology, we slightly refined TAL to increase its expressiveness and resolve ambiguities that could otherwise affect ontological assertions and inferencing. TAL's threat actor types and their associated defining attributes are shown in Table 1. The table's key takeaways are: TAL comprises twenty-one unique threat actor type categories and their associated characteristics based on eight attributes. The motivation attribute was added to the library in later work [3]. The shaded cells in the second column of Table 1 refer to either minor nonbreaking attribute modifications that resolve ambiguity concerning their ontological use, or attribute updates that allow for more flexible use. For instance, the individualistic motivation *Personal Financial Gain* has been replaced with *Financial Gain* to allow more flexible characterization, meaning that the property can now be used to characterize groups and not only individuals, such as organized cyber crime groups that operate mainly for profit, indicating financially motivated actors.

A high-level illustration of the ontology is presented in Figure III.9. The threat actor type and characterization attribute classes enumerate possible values using individuals (instances). For example, the visibility attribute comprises four individuals that define different levels of visibility: clandestine, covert,

opportunistic, and overt.

Object properties relate individuals to individuals. For example, an individual (object) that describes an adversarial operation can have a relationship to a motivation that is believed to influence the attack, such as the desire to achieve *dominance*. This can be expressed using the object property *hasDefiningMotivation*, deriving a semantic triple (*subject-hasDefiningMotivation-dominance*).

In addition, the ontology can automatically infer threat actor types, decreasing the human biases entailed in traditional manual classification and decision-making processes, by capturing the existing domain knowledge within ontology expressions (axioms) that characterize threat actor types based on combinations of the attributes mentioned earlier. An example expression that captures the combination of attributes comprising a nation-state-backed actor (government cyberwarrior based on TAL) is shown below in Manchester syntax.

```
((hasVisibilityAttribute some Visibility) or
(hasVisibilityAttribute value visibility:dontCare))
 and ((hasObjectiveAttribute value objective:damage) or
 (hasObjectiveAttribute value objective:deny) or
 (hasObjectiveAttribute value objective:destroy))
 and ((hasOutcomeAttribute value outcome:damage) or
 (hasOutcomeAttribute value outcome:embarrassment))
 and (hasAccessAttribute value access:external)
 and (hasDefiningMotivationAttribute value motivation:dominance
     )
 and (hasLimitsAttribute value limits:extraLegalMajor)
 and (hasResourcesAttribute value resources:government)
 and (hasSkillsAttribute value skills:adept)
```

Objects with populated attributes that fulfill expression requirements (equivalency) are classified as threat actor types in an automated manner near real-time by a description logics reasoner. As demonstrated in Section III.5, polymorphic threat groups can be attributed to more than one threat actor type, compared to traditional enumerative approaches that use mutually exclusive lists and lead to a contextual loss. The suggested approach does not prohibit an analyst from manually classifying a threat actor as a specific type or populating other attributes (open world assumption). Changes to the defining characterizations of threat actor types can be reflected by updating the ontology expressions. To enable temporality, the characterization attributes of a threat actor instance are populated using an individual object (instance) that connects with other related instances (e.g., malicious activity or identity) using relationships (Figure III.7). Temporality-based knowledge representation can justifiably reflect shifts and polymorphism in adversarial behavior.

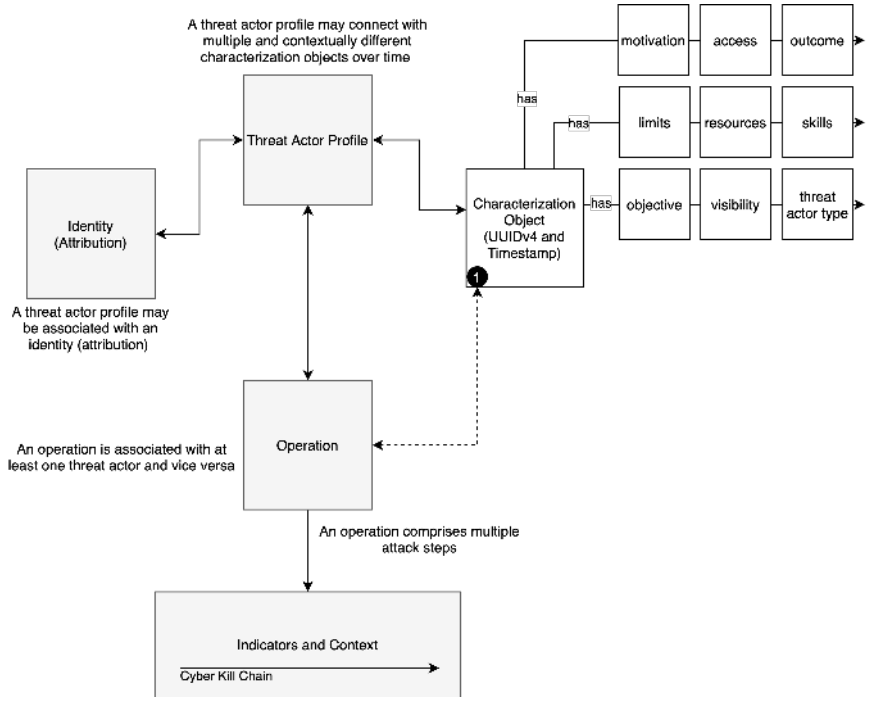

Figure III.7: Temporality Enhanced Semantic Modeling of Threat Actor Polymorphism

III.5 The Lazarus Group Use Case

In this section, we utilize the ontology presented in Section III.4 to model the Lazarus Group for the purpose of inferring threat actor types automatically. We demonstrate how a standardized set of characterization attributes for describing adversary capability and behavior makes cyber threat intelligence more contextual and queryable, and makes it possible to derive new information at machine speed by utilizing a reasoner. We apply a top-down modeling approach to open-source information about operations believed to have been conducted by the Lazarus Group. Even though an attribution of high confidence has been achieved and the capabilities and sophistication of the Lazarus Group are known, we characterize the operations (use cases) based on their individual characteristics. A top-down modeling approach uses existing knowledge and historical data to create a threat actor profile and is more accurate and contextual than a bottom-up approach which derives intelligence from early-stage ongoing analyses of cyber attacks. Nevertheless, both modeling methods should follow an evidence-based approach by establishing direct relationships between the characterization attributes and the instances of operations the information has been derived for robust, explicable, and temporal-enabled threat intelligence.

According to the MITRE ATT&CK Groups knowledge base[6]:

> *"Lazarus Group is a threat group that has been attributed to the North Korean government. North Korean groups are known to have significant overlap, and the name Lazarus Group is known to encompass a broad range of activity. Some organizations use the name Lazarus group to refer to any activity attributed to North Korea, whereas other organizations track North Korean clusters or groups such as Bluenoroff, APT37, and APT38 separately."*

According to the Council on Foreign Relations[7]:

> *"Lazarus Group targets and compromises entities primarily in South Korea and South Korean interests for espionage, disruption, and destruction. It has also been known to conduct cyber operations for financial gain, including targeting cryptocurrency exchanges."*

The descriptions above are indicative of a polymorphic threat. Based on TAL, an ontological equivalency expression of a nation-state threat actor (government cyberwarrior) identifies the following characteristics:

- (*access*→*external*)
- (*visibility*→*any-opportunistically*)
- (*objective*→*deny-destroy-damage*)
- (*limits*→*extra-legal, major*)
- (*outcome*→*damage, embarrassment*)
- (*defining motivation*→*dominance*)
- (*skills*→*adept*)
- (*resources*→*government*)

Establishing formal threat actor type definitions using a set of machine-readable characterization attributes equips defenders with a queryable representation that can derive explicable intelligence.

The Lazarus Group is known to have been active for more than a decade and is an example of an adversary that has exhibited polymorphism and increased operational sophistication over time. The nation-state-backed group has engaged in multiple cyber espionage, destructive, disruptive, and financially motivated operations. For example, the DarkSeoul attack on March 20, 2013, targeted South Korean news agencies and banks, causing significant damage to the affected entities by wiping the hard drives of tens of thousands of computers.

[6] https://attack.mitre.org/groups/G0032/
[7] https://www.cfr.org/cyber-operations/lazarus-group

At an early stage, Symantec stated that the actual motives for the attacks were unclear and added that they might be part of either a clandestine attack or the work of *nationalistic hacktivists* taking issues into their own hands in response to political tensions on the Korean Peninsula [7]. In a report [17], McAfee, after analysis, remarked that an attack which was initially perceived as an unsophisticated incident of *cyber vandalism* or *hacktivism* had actually grown out of a sophisticated multi-year covert cyber espionage campaign that this time was indeed intended to damage, cause disruption, and potentially harvest information. Table 1 identifies the defining characteristics of a cyber vandal and radical activist according to TAL.

The threat actors NewRomanic Cyber Army Team and Whois Team, who claimed responsibility for the attacks in South Korea, were later discovered to be a fabrication to mask the real source of the attack. In addition, Marpaung and Lee explained that DarkSeoul was a low-tech threat compared to advanced persistent threats that nation state groups typically perform [9]. Often the level of an attack's sophistication is inversely proportional to its magnitude [10]. For example, an attack like Stuxnet that is narrowly targeted and technologically sophisticated indicates an operation by a highly organized group with the skills and resources to develop a persistent and destructive attack.

By structuring the information about the DarkSeoul attack, the following characterization attributes emerge. The threat actor was external to the targeted entities (*access*→*external*) and conducted a large-scale covert operation (*visibility*→*covert*) which caused destruction, disruption, and possibly harvested information (*objective*→*destroy, damage,* and maybe *copy*). Based on the attack type and impact, we can conclude that the actor took no account of the law (*limits*→*extra-legal major*) and that its primary goal was large-scale data destruction with a sequential impact on the affected entities' operations (*outcome*→*damage*). This type of attack reflects a motivation to achieve dominance over another party, or as in this case, over another nation (*defining motivation*→*dominance*). Furthermore, what was initially perceived as an unsophisticated attack due to the raw destructive nature of the payload was, in fact, a coordinated strike against multiple entities delivered with precision and planning commonly associated with state-sponsored intrusion campaigns [17] (*skills*→*adept*), (*resources*→*government*). Based on the above characterization, a reasoner would infer that a *government cyberwarrior* conducted the operation, otherwise known as a *nation state* threat actor. It is worth noting that the contextual characterization of the DarkSeoul attack in this particular case takes into account information about a set of individual attacks all described in one object, thus indicating a relatively high-level sophistication, which in turn is a factor for estimating the skills and resources required for conducting the attacks. Exemplifying each incident separately would populate objects that a reasoner would infer as threat actor type *(cyber) vandal*. The attributes such as motivation, outcome, objectives, and visibility highly overlap between the *vandal* and *government cyberwarrior* (nation state) categories. Other attributes such as skills, resources, and limits are dissimilar and annotate the differences in capability between the two types. The attribution of the DarkSeoul attack

confirmed that it was planned and executed by a known *nation-state* threat actor.

Another similar incident occurred on June 25, 2013, on the 63rd anniversary of the start of the Korean War (1950–1953), which resulted in the division of the Korean peninsula. On that day, multiple attacks reminiscent of *nationalistic hacktivism*, a type of patriotic activism, targeted the Blue House, government ministries, and media by defacing web pages, stealing data, and corrupting servers. One of the distributed denial-of-service (DDoS) attacks observed against the South Korean government websites was directly linked to malware used in the DarkSeoul attack [6]. The ontology in Section III.4 does not account for a *nationalistic hacktivist* threat actor type that would ideally characterize this operation's actor. The defining attributes of each threat actor type describe their subtle differences. For example, even though the characterization attributes of the *nationalistic hacktivist* type would highly overlap with the *radical activist* type in terms of outcomes and objectives, nationalistic hacktivists are mainly motivated by the desire to achieve dominance over another nation because of their loyalty and strong devotion to their own nation or the leaders of the nation. In contrast, a radical activist operates for more ideological and political reasons to replace the fundamental principles of a society or a political system. In addition, nationalistic hacktivists would be resource-constrained compared to a nation-state-backed group. As explained in Section III.3, the definition of new actor types and updating existing ones should be a standards-based task where the security community agrees on explainable characterization attribute-based descriptions for promoting and facilitating universal adoption.

In November 2014, Sony Pictures Entertainment (SPE) was attacked with malware resulting in information theft which was later used for extortion regarding canceling the release of a film depicting an assassination plot against North Korean leader Kim Jong Un. The stolen data included employee personal information, company emails, usernames and passwords, details of SPE's internal IT infrastructure, and unreleased movies. In addition, the attackers succeeded in rendering thousands of computers inoperable by deleting the master file table and the master boot record from hard drives [20]. The perpetrators identified themselves as Guardians of Peace (GOP). The attack, which was initially believed to be the work of a *hacktivist group* or *disgruntled insiders*, was later attributed to Lazarus Group [16]. Based on available information, we characterize the operation and derive the following attributes. The Sony incident was a covert operation (*visibility*→*covert*) planned and executed by an unknown external group (*access*→*external*) that caused theft of information and damage to assets (*objective*→*copy, damage, destroy*). The stolen information was used to hurt the company's image and resulted in significant financial losses (*outcome*→*damage, embarrassment*). The extortion demands, in addition to threatening emails sent to Sony employees, reflected a threat actor who takes no account of the law (*limits*→*extra-legal, major*) and an actor who attempts to achieve dominance through its actions (*defining motivation*→*dominance*). In addition, the threat actor demonstrated considerable resources and advanced skills, as indicated by its persistence in Sony's network and the significant

losses suffered (*skills→adept*), (*resources→at least organization*). Based on the above characterization, a reasoner would infer that the populated attributes are equivalent to *government cyberwarrior* or otherwise known as *nation state* threat actor type. Nevertheless, the attack could also be understood as a form of *nationalistic hacktivism* because of its context. Interestingly, in the early stage of the attack and before the explicit demand to withdraw the movie's theatrical release, some of the targeted high-ranking Sony employees received compensation requests from the attackers for the damage they had suffered [20]. This could indicate a personal financial motivation, irrespective of the group's primary goal.

The Lazarus Group, being polymorphic, has also been observed to be financially motivated and has demonstrated highly organized and sophisticated cyber criminal behavior by penetrating targets with large financial streams. According to Kaspersky [8], Lazarus Group operations are expensive, and financially motivated attacks could be a way to better finance them. Chanlett-Avery et al. emphasized that Lazarus Group engages in financially motivated attacks to raise revenue for the regime in response to sanctions imposed by the United States and the United Nations Security Council as a reaction to North Korea's weapons of mass destruction and ballistic missile programs, as well as human rights abuse [5].

Temporality-based semantic representation and inference provide more complete, queryable, and explainable intelligence and a certain extent of automation in intelligence generation with respect to how threat actors evolve into new behaviors. Based on the queries that an organization wants to answer, the characterization attributes and inferred information (instances) can be used to derive highly relevant and contextual cyber threat intelligence. Furthermore, universally agreed unambiguous definitions and vocabularies enable more robust information sharing.

As illustrated by Figure III.8, the evidence indicates that Lazarus Group is polymorphic and, through its operations, has exhibited behavior and capability aligned with organized cyber crime, nationalistic hacktivists, cyber vandals, and nation-state-backed entities.

III.6 Conclusion

Threat actors are becoming increasingly sophisticated and polymorphic. To understand those hybridized threats, defenders seek timely, accurate, relevant, and actionable threat intelligence for anticipatory threat reduction. Today's threat intelligence tends to be ambiguous and inadequately structured to track and demystify changes in the behavior of actors over time, such as new goals, motivations, and related operations and TTPs. Threat actors have an asymmetric information advantage over defenders. Before executing a targeted attack, they are well aware of the profiles, infrastructures, systems, and applications of their victims. This work laid the foundation for generating highly contextual, explicable, processable, and shareable threat actor intelligence that can accurately capture, interpret, and explain changes in threat actor behavior

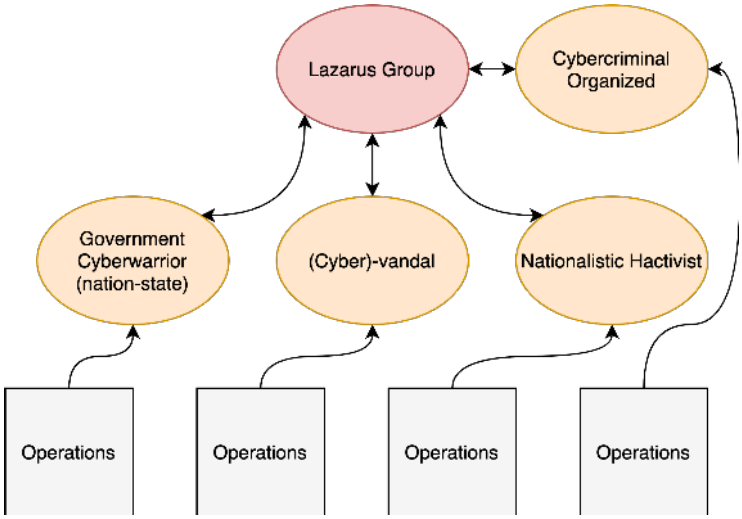

Figure III.8: Polymorphism of Lazarus Group

and their polymorphism over time. In particular, we demonstrated how a set of characterization attributes can enrich threat actor information and how, in combination, can enumerate their type. By encapsulating this knowledge within an ontology, we demonstrated how a perpetrator's nature could be inferred automatically using deductive reasoning and withhold the relations/semantics that justify the inference.

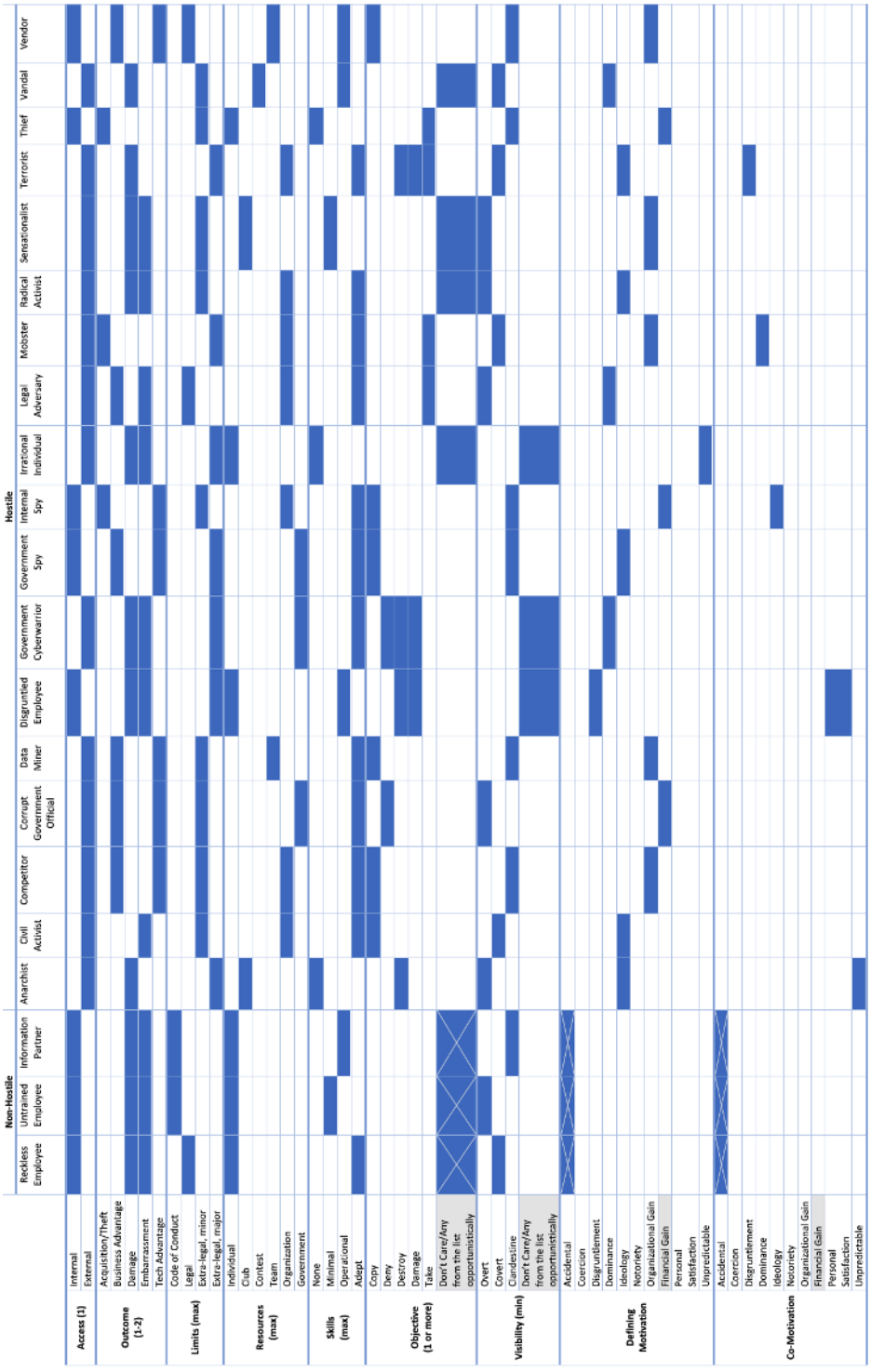

Table III.1: Threat Agent Library

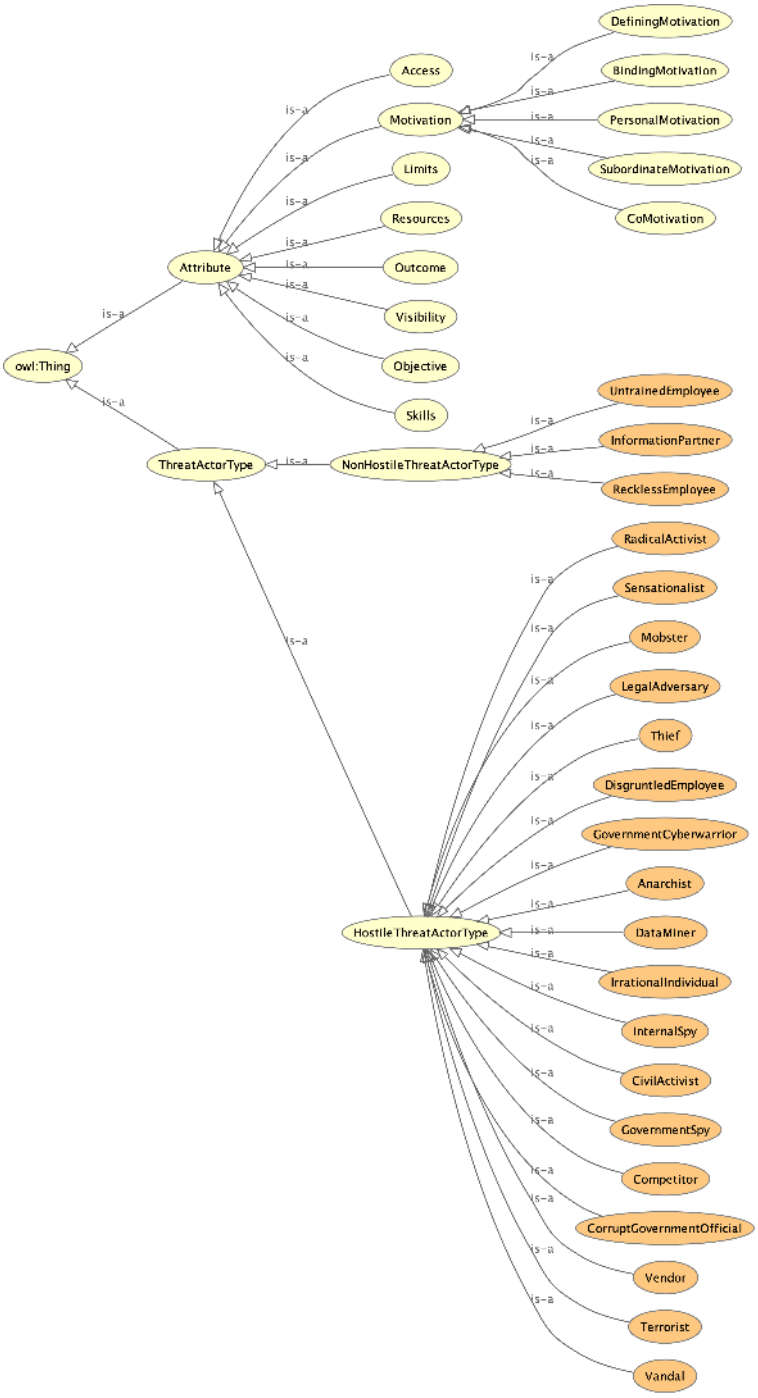

Figure III.9: High-level Representation of Ontology Classes and Associated Individuals

Acknowledgements. This research work was supported by the research project CyberHunt (Grant No. 303585) funded by the Research Council of Norway. The authors would like to express their gratitude to Mr. Paul Patrick from DarkLight Inc. for providing comments that helped improve the manuscript.

References

[1] Alexander, O., Belisle, M., and Steele, J. *MITRE ATT&CK® for Industrial Control Systems: Design and Philosophy*. Tech. rep. MITRE Corporation, 2020.

[2] Bianco, D. *The Pyramid of Pain*. Accessed: Jul. 2020. [Online]. Available: http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html. 2014.

[3] Casey, T. *Understanding Cyber Threat Motivations to Improve Defense*. Tech. rep. Intel Corporation, 2015.

[4] Casey, T. *Threat Agent Library Helps Identify Information Security Risks*. Tech. rep. Intel Corporation, 2007.

[5] Chanlett-Avery, E. et al. *North Korean Cyber Capabilities: In Brief*. Tech. rep. Congressional Research Service, 2017.

[6] Johnson, A. L. *Four Years of DarkSeoul Cyberattacks Against South Korea Continue on Anniversary of Korean War*. Accessed: Aug. 2020. [Online]. Available: https://community.broadcom.com/symantecenterprise/communities/community-home/librarydocuments/viewdocument?DocumentKey=edd5c93e-7160-4bf2-a15c-f1c024feb0d7&CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68&tab=librarydocuments. 2013.

[7] Johnson, A. L. *South Korean Banks and Broadcasting Organizations Suffer Major Damage from Cyberattack*. Accessed: Aug. 2020. [Online]. Available: https://community.broadcom.com/symantecenterprise/communities/community-home/librarydocuments/viewdocument?DocumentKey=6859f4a7-d5c2-4a81-bbed-dfa70470e9db&CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68&tab=librarydocuments. 2013.

[8] Kaspersky Lab. *Lazarus Under The Hood*. Tech. rep. Accessed: Aug. 2020. [Online]. Available: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180244/Lazarus_Under_The_Hood_PDF_final.pdf. Kaspersky, 2017.

[9] Marpaung, J. and Lee, H. "Dark Seoul Cyber Attack: Could it be worse?" In: Proceedings of the Conference of Indonesian Student Association in Korea. 2013.

[10] Martin, D. *Tracing the lineage of DarkSeoul*. Tech. rep. SANS Institute, 2015.

[11] Meyers, A. *Meet CrowdStrike's Adversary of the Month for April: STARDUST CHOLLIMA*. Accessed: Aug. 2020. [Online]. Available: https://www.crowdstrike.com/blog/meet-crowdstrikes-adversary-of-the-month-for-april-stardust-chollima/. 2018.

[12] MITRE. *Adversarial Tactics, Techniques Common Knowledge (ATT&CK)*. Accessed: Jul. 2020. [Online]. Available: https://attack.mitre.org/.

[13] Motik, B., Patel-Schneider, P., and Parsia, B. *OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition)*. W3C Recommendation. W3C, Dec. 2012.

[14] Nickels, K. *Getting Started with ATT&CK: Threat Intelligence*. Accessed: Jul. 2020. [Online]. Available: https://medium.com/mitre-attack/getting-started-with-attack-cti-4eb205be4b2f/. 2019.

[15] OASIS. *OASIS Structured Threat Information Expression (STIX™) Version 2.1*. Standard. OASIS, 2021.

[16] *Operation Blockbuster: Unraveling the Long Thread of the Sony Attack*. Tech. rep. Novetta, 2016.

[17] Sherstobitoff, R., Liba, I., and Walter, J. *Dissecting Operation Troy: Cyberespionage in South Korea*. Tech. rep. McAfee, 2013.

[18] Symantec DeepSight Adversary Intelligence Team. *Waterbug: Espionage Group Rolls Out Brand-New Toolset in Attacks Against Governments*. Accessed: Jun. 2020. [Online]. Available: https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/waterbug-espionage-governments. 2019.

[19] ThaiCERT. *Threat Group Cards: A Threat Actor Encyclopedia*. Tech. rep. Accessed: Aug. 2020 [Online]. Available: https://www.thaicert.or.th/downloads/files/Threat_Group_Cards_v2.0.pdf. Electronic Transactions Development Agency, 2020.

[20] United States Department of Justice. *North Korean Regime-Backed Programmer Charged in Conspiracy to Conduct Multiple Cyberattacks and Intrusions*. Accessed: Aug. 2020. [Online]. Available: https://www.justice.gov/usao-cdca/pr/north-korean-regime-backed-programmer-charged-conspiracy-conduct-multiple-cyberattacks. 2018.

[21] Wagner, C. et al. "Misp: The design and implementation of a collaborative threat intelligence sharing platform". In: *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*. 2016, pp. 49–56.

Paper IV

A Nonproprietary Language for the Command and Control of Cyber Defenses - OpenC2

Vasileios Mavroeidis[1], Joe Brule[2]

Published in: *Computers & Security*, Volume 97, October 2020, Article 101999, DOI: 10.1016/j.cose.2020.101999, ELSEVIER.

Abstract

The fact that cyber attacks are getting increasingly sophisticated and performed at machine speed motivated the development of OpenC2. This paper presents Open Command and Control (OpenC2), a suite of specifications that enable command and control of cyber defense systems and components at machine speed and in a manner that is agnostic of the underlying technologies utilized or of any other aspects of particular implementations. OpenC2 provides the means to introduce standardized interfaces to cyber defense systems, enabling interoperability and allowing seamless integration, communication, and operation between decoupled blocks that perform cyber defense functions. The suit of specifications includes a semantic language that enables machine-to-machine communication for purposes of command and control of cyber defense components, actuator profiles that specify the subset of the OpenC2 language and may extend it in the context of specific cyber defense functions, and transfer specifications that utilize existing protocols and standards to implement OpenC2 in particular environments. Fundamentally, OpenC2 addresses the acting part of the Integrated Adaptive Cyber Defense (IACD) framework and is designed to be technology agnostic, concise, abstract, and extensible. Ultimately, OpenC2 is a building block for enabling coordinated defense in cyber-relevant time, shifting traditional monolithic cyber response approaches to more granular, flexible, and adaptive.

[1]University of Oslo, Oslo, Norway, vasileim@ifi.uio.no
[2]National Security Agency, USA, jmbrule@radium.ncsc.mil

IV.1 Introduction

The attack surface of cyber systems is relative to their complexity, functionality, and connectivity. Adversaries and their tactics, techniques, and procedures have become increasingly sophisticated, well-funded, and can operate at machine speed. The impact of a cyber attack can be detrimental to the well-being of a nation and the longevity of an organization whether is derived from significant financial losses and loss of intellectual property or more severe effects, such as failures in functions of critical national infrastructure that can rapidly lead to massive disruption in society and even loss of lives.

Cyber defense systems mostly operate in isolation and are often statically configured, resulting in poorly integrated cybersecurity operations. As a result, investigating and responding to cybersecurity incidents, especially large-scale and sophisticated, is a non-timely and cumbersome process that provides an asymmetric time advantage to adversaries for performing and maneuvering their attacks successfully.

Remediation and mitigation plans against cyber attacks comprise coordinated action-sets that work synergistically by utilizing multiple technologies. The integration of different cyber defense components and technologies can be expensive and requires customized communication interfaces. Such customized interfaces come in the form of application programming interfaces (APIs) that are proprietary to the parent technology. Furthermore, the functional blocks within a product may be tightly coupled with other functions and may not be directly accessible by an API, reducing the product's dynamic and integration flexibility with other tools. Overall, an integration would require middleware that translates, stores, and forwards messages between technologies. In some sense, nowadays, this is achieved by different orchestration platforms that adopt a plug and play architecture where different technologies can be integrated and interoperate. Such solutions are also difficult to scale since they depend on the availability of API hooks for the different products that the platform supports, or require developing custom interfaces for products that are not currently supported by the platform. Further, adopting technologies from multiple vendors undeniably introduces an extra layer of security. For example, a zero-day vulnerability that affects a specific firewall series by allowing remote code execution should not affect every firewall in an organization's infrastructure; thus, firewall-vendor diversity is desirable. Vendor diversity adds strength to an organization's infrastructure but also has its limitations. The defense is most of the times non-integrated, and that makes device management more complicated and costly.

Introducing standardized function-centric interfaces for command and control enhances the ability to technologically diversify, makes device management less complicated, and simplifies integration. Also, coordinated cyber defense in cyber-relevant time requires machine-to-machine communication. A strategy that decouples the functional blocks within a cyber defense system and the definition of standardized interfaces, through a common language, for seamless communication would allow flexible integration of cyber defense components and permit incremental upgrades. An open standard language for the command

and control of cyber defense systems enables interoperability between different technologies and decreases the response time to cyber attacks. For instance, an ongoing incident at an organization with multiple disparate branches and centralized security operations might require updating numerous firewalls from different vendors in many locations. This can be addressed in multiple ways, such as either manually configuring each firewall by remotely accessing the target devices and updating the required rules, or utilizing a GUI-based solution that allows the configuration of different devices from different vendors with the use of APIs. Standardized interfaces that utilize a common language for command and control would allow updating the firewalls in machine time by issuing only a single command. The command itself should be agnostic of the underlying device that will consume it but it should be function-relevant. The device itself remains responsible for understanding the command issued in a common language and format, and executing it.

The aforementioned shortcomings in traditional cyber defense approaches and the identified requirements for enabling coordinated cyber response in cyber-relevant time motivated the development of OpenC2. Open Command and Control (OpenC2) is a suite of specifications that enable command and control of cyber defense systems and components at machine speed and in a manner that is agnostic of the overall underlying technologies utilized. OpenC2 aims to standardize the way cyber defense systems and functions communicate and consequently interoperate in a way that security automation and orchestration become feasible and less complex to achieve.

Concisely, the suite of specifications includes the language specification, actuator profiles, and transfer specifications. The OpenC2 language specification provides the semantics for the essential elements of the language, the structure for commands and responses, and defines the proper compositions and data types for the language elements that represent the command or response. OpenC2 actuator profiles extend and specify subsets of the OpenC2 language relevant to particular cyber defense functions. Examples of cyber defense functions include stateless and stateful packet filtering. Actuator profiles also provide the appropriate conformance requirements and recommendations for enabling interoperability between different technologies. The OpenC2 transfer specifications utilize existing protocols and standards for encoding and communicating OpenC2 messages securely.

The rest of the paper introduces and delves into the mechanics of OpenC2 and presents a use case implementation.

IV.2 Open Command and Control - OpenC2

OpenC2 [11] is developed by a domain-expert technical committee within the OASIS international standards body. The National Security Agency of the United States Department of Defense, the Bank of America, and the University of Oslo, to name a few, are among many organizations and agencies globally that support the effort and offer expertise for its successful development (OpenC2

involves members from industry, government, and academia). The Organization for the Advancement of Structured Information Standards [1] (OASIS) is a global nonprofit consortium that drives the development, convergence, and adoption of many open standards for the global information society.

IV.2.1 OpenC2 Scope

Real-time detection and mitigation of threats at every tier in every cyber environment require the integration, synchronization, and automation of sensing, sense-making, decision-making, and acting capabilities by secure automated orchestration and the development of messaging and C2 infrastructure standards [7].

A high-level decomposition of the functional blocks within a cyber-defense ecosystem or a single advanced system includes:

- Sensing: collection of data from sensors with the intent to provide awareness.
- Sense-making: analytics to provide understanding in a particular context and the current state based on the collected data.
- Decision-making: selection of response actions relevant to the current state.
- Responding/Acting: execution of a selected course of action to mitigate, remediate, or further investigate a situation as indicated by a response plan. This can be a mix of automated and manual actions integrating human in the loop.
- Message Fabric: assured communications infrastructure to ensure a standard communication medium for all the technologies involved, and seamless execution of commands in a timely manner by authenticated and authorized entities.

The above, also known as Integrated Adaptive Cyber Defense [14], is analogous to the classic OODA control loop (observe, orient, decide, and act) but tailored to cybersecurity operations with the purpose of promoting and leveraging automation in cyber defense. The OODA loop is a military approach for disrupting effectively and timely adversarial operations by iteratively collecting and processing information for making the right decisions and acting at a faster pace than the adversaries.

OpenC2 addresses the response (acting) segment of cyber defense, also known as course of action. Course of action refers to measures that can be taken to prevent or respond to attacks [10]. The defined language enables unambiguous machine-to-machine communication and is agnostic of any particular transfer or transport protocol, and information assurance implementation. Other aspects

[1] https://www.oasis-open.org/

of coordinated cyber response such as sensing, sense-making (analytics), and selecting appropriate courses of action are beyond the scope of OpenC2 but are required to enable coordinated cyber response in cyber-relevant time. Thus, OpenC2 assumes that the rest of the functional blocks are in place.

IV.2.2 OpenC2 Terminology

This section elucidates terminology pertinent to OpenC2.

- Action: a single task to be performed. An action (e.g., deny, update, contain, restart) is an instruction from a producer to a consumer and is executed by an actuator (e.g., stateful or stateless packet filter).
- Target: the object of the action. An action is performed on a target (e.g., IP address, file, process, device).
- Argument: a property that provides additional granularity with respect to how, when, and where to perform a command (e.g., date, time, periodicity, duration, specific interface). Arguments are context-dependant (action-target pair dependant).
- Specifier: a property or field that identifies a target or actuator to some level of precision.
- Actuator: the function performed by the consumer that executes the command (e.g., stateless or stateful packet filtering). An actuator is defined within the context of an actuator profile.
- Actuator Profile: a subset of the OpenC2 language relevant to a specific cyber defense function. An actuator profile may extend the OpenC2 language by defining targets, command arguments, and specifiers that are relevant and/or unique to a specific actuator function.
- Command: a message defined by an action-target pair and possibly additional arguments and specifiers that is sent from a producer, received by a consumer, and executed by an actuator.
- Response: a message from a consumer to a producer acknowledging a command or returning the requested resources or status based on a previously received command.
- Message: a content- and transport-independent set of elements conveyed between producers and consumers.
- Producer: an entity (device, application, functional block) that generates and sends commands. Note that a single entity can have both producer and consumer capabilities.

- Consumer: an entity (device, application, functional block) that receives and possibly acts upon commands. Note that a single entity can have both consumer and producer capabilities and support multiple actuator functions.

IV.2.3 OpenC2 Overview

OpenC2 aims to enable coordinated defense in cyber-relevant time between decoupled blocks that perform cyber defense functions. The assumption that underlies the design of OpenC2 is that the sensing and analytics for sense-making have been provisioned and the decision to act has been made. OpenC2 was designed based on the following four principles:

- Technology Agnostic: the OpenC2 language defines a set of unbiased abstract and atomic cyber defense actions, enabling interoperability among cyber defense systems independently of any other aspects of the underlying implementations.
- Concise: the OpenC2 language is minimal, focusing only on the essential information needed to derive targeted cyber defense actions. The language is designed to provide minimum overhead in the communication of OpenC2 messages and is appropriate for network constrained environments.
- Abstract: OpenC2 commands and responses are defined abstractly and can be encoded and transferred via multiple schemes as dictated by the needs of different implementation environments.
- Extensible: the OpenC2 language should evolve alongside cyber defense technologies. Supported by the aforementioned design principles OpenC2 can be extended for introducing new cyber defense functionality.

OpenC2 uses a request-response paradigm where a command is generated and encoded by a producer and transferred to a consumer using a secure transfer protocol (Figure IV.1).

The consumer normally executes the received command and can respond to a producer with the status of the execution and other requested information. A producer can adjust its behavior based on the capability (supported features) of a consumer. In particular, a producer can request details about which versions of OpenC2 language, actuator profiles, and action-target pairs a consumer supports. The capability definitions can be easily extended in a non-centralized manner, allowing standard and non-standard capabilities to be defined with semantic and syntactic rigor. It needs to be emphasized that not all targets are meaningful in the context of a specific action. Although a command such as "update ipv4_connection" may be syntactically valid, the combination does not reflect an operation supported by a particular actuator profile. For example, the actuator profile for Stateless Packet Filtering (SLPF) version 1.0 defines only one target, namely "file", relevant to the action "update" (Figure IV.2). Consumers and

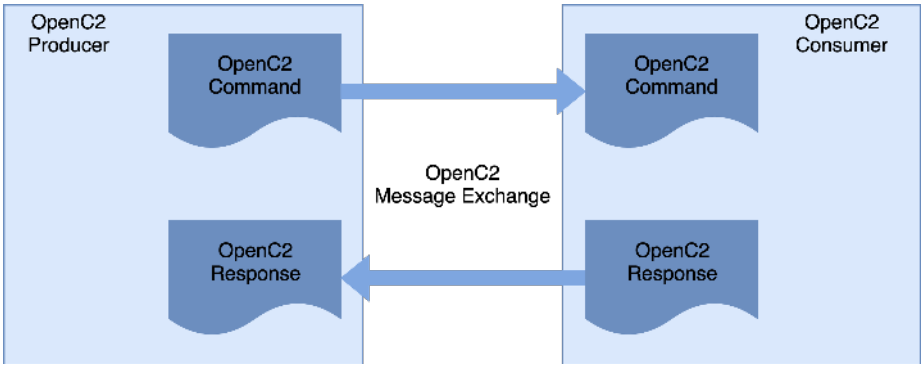

Figure IV.1: OpenC2 Message Exchange

producers must satisfy the requirements specified in the language specification and relevant actuator profile(s) to claim conformance. Compliance with the conformance clauses enables interoperability, the ability of a system to operate in conjunction with other systems.

Command Matrix					
	Allow	Deny	Query	Delete	Update
ipv4_connection	●	●			
ipv6_connection	●	●			
ipv4_net	●	●			
ipv6_net	●	●			
features			●		
slpf/rule_number				●	
file					●

Figure IV.2: OpenC2 SLPF Command Matrix

IV.2.4 OpenC2-enabled Functionality and Integration

As OpenC2 evolves and matures through additions, updates, adoption, and testing, we expect more vendors to introduce native support. A native OpenC2 interface is a capability that allows systems and functions to be called and managed directly by other systems or functions using the OpenC2 language, enabling interoperability and eliminating the need for any middleware. Cyber defense systems that natively support OpenC2 can produce or consume OpenC2 commands without the need of any translation service for interpreting the commands to the consumer proprietary language and syntax. This approach shifts the interface engineering complexity, conformance testing, and maintenance

at the vendors' side compared to approaches that demand middleware to perform translations, such as proxies that are also most of the time community-based software efforts. An example architecture with OpenC2 interfaces is presented in Figure IV.3, where producers and consumers communicate natively, eliminating the multidimensional complexity and the computing and network overhead any additional translation technology would introduce.

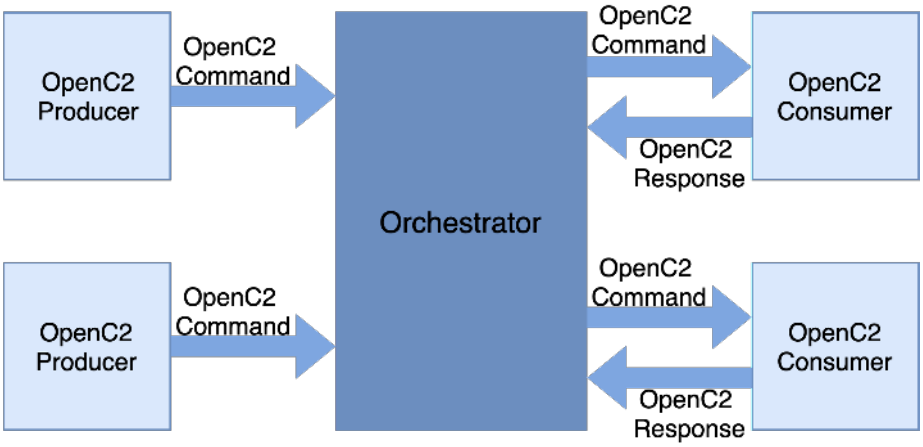

Figure IV.3: Architecture with Native OpenC2 Interfaces

As observed in Figure IV.3, an orchestrator serves as a mission manager that issues or forwards OpenC2 commands to consumer actuators. Defining orchestrator architectures or technology is out of the scope of OpenC2. However, an orchestrator should be able to send, receive, and keep track of OpenC2 commands and responses, register and authenticate devices and functions, deal with certificate management, and may support multiple serializations and transfer protocols. In cases where cyber defense systems do not natively support OpenC2, integration is achieved by introducing middleware technology, such as a proxy, that does the appropriate translation/mapping from OpenC2 to the relevant vendor-proprietary notation and maybe act as a messaging infrastructure (Figure IV.4). The positioning of the technologies in Figures IV.3 and IV.4 is notional for aiding the understanding of the reader and do not represent factual architectures.

IV.2.5 OpenC2 Serialization

Serialization is the process of converting an object into stream of bytes so that it can be stored or transferred over a network. Its main purpose is to save the state of an object (in our case, an OpenC2 command or response) in a standard format that can be transferred, reconstructed, and understood by any other application or system that supports the same serialization. The OpenC2 language is agnostic of any particular serialization format; however, implementations must support JSON (JavaScript Object Notation) in accordance with RFC 7493 [2]

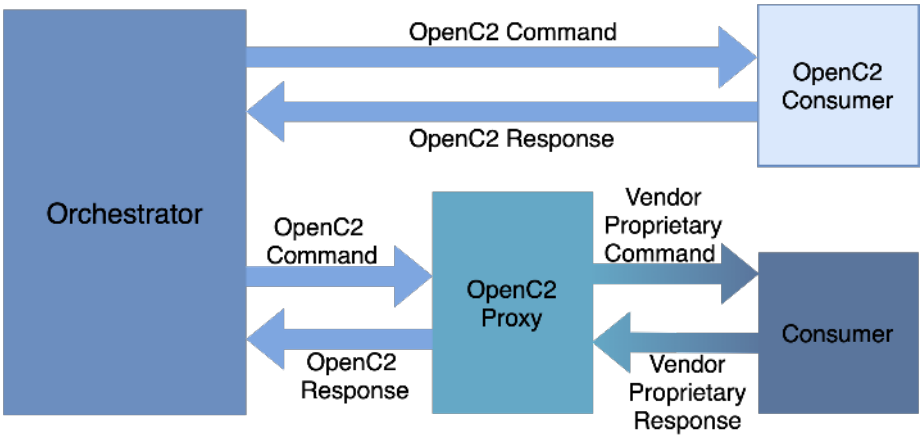

Figure IV.4: Architecture with OpenC2 Proxy (Middleware)

and additional requirements as to how OpenC2 data types are represented in JSON. For instance, an OpenC2 data type "IPv4-Addr" that specifies an Internet Protocol address version 4 must be a JSON string containing the "dotted-quad" representation of an IPv4 address as specified in RFC 2673, Section 3.2 [4]. An OpenC2 data type "MAC-Addr" that represents a media access control address must be a JSON string containing the text representation of a MAC address in colon hexadecimal format as defined in [8]. The syntax of an OpenC2 command in JSON is presented right below.

```
{
  "action": <action type>,
  "target": {
    <target type>: {
      <target specifiers as key:value pairs>}},
  "args": {
    <general command arguments as key:value pairs>,
    <actuator type>: {
      <actuator-relevant command arguments as key:value pairs
          >}},
  "actuator": {
    <actuator type>: {
      <actuator specifiers as key:value pairs>}}
}
```

Reflecting on a use case where a sandbox and a firewall are part of an orchestration process (using a security playbook), identified malicious C2 infrastructure, such as IPs related to malware, could be forwarded to the orchestrator and from the orchestrator to the consumer. The orchestrator knowing the actuator's capability, which in our case is a firewall device, will populate and forward

precise OpenC2 commands that can block the target IPs. The commands are expressed in JSON, transferred as 1's and 0's in the wire, and reconstructed back to JSON for processing and consumption by the indicated actuator. An OpenC2 command relevant to the described use case is presented below.

```
{
   "action":"deny",
   "target":{
      "ipv4_net":"malicious C2"
   },
   "args":{
      "response_requested":"complete"
   },
   "actuator":{
      "slpf":{
         "hostname":"firewall-01"
      }
   }
}
```

The command defines a "deny" action that blocks an IPv4 address (target) expressed in CIDR notation (classless inter-domain routing), requests a response message about the status and maybe the results of the performed command by the actuator, and specifies the actuator that will perform the requested action. A response indicating that the actuator successfully issued the command is presented below, where according to the OpenC2 Language Specification version 1.0 in a response message, a response code is required, and a status text is optional.

```
{
  "status": 200,
  "status_text": "OK"
}
```

Consumers support one or more actuator profiles and possibly more than one serialization formats (e.g., JSON and CBOR) based on their capability. The command above is populated based on the Stateless Packet Filtering (SLPF) version 1.0 actuator profile supported by "firewall-01". On that basis, both consumers and producers should fulfill the actuator-specific requirements annotated in the relevant specification for successfully inter-communicating and interpreting the messages. For example, when the mask of an "ipv4_net" target is unspecified, it must be treated as a single IPv4 address. Additionally, the address range specified in the "ipv4_net" must be treated as a source or destination address. The exact serialization format used for message exchange is annotated and included in the header of a message.

IV.2.6 OpenC2 Language

The OpenC2 language is described in the Open Command and Control (OpenC2) Language Specification document, currently, in version 1.0 [13]. The language, as described in Section IV.2.3, is designed to be technology agnostic, concise, abstract and extensible, and is used to compose messages for command and control of cyber defense systems and components. The OpenC2 language is used in conjunction with OpenC2 actuator profiles that extend the language in the context of particular cyber defense functions (discussed in Section IV.2.7), and OpenC2 transfer specifications that provide guidance on how OpenC2 messages should be transferred over specific transfer protocols (discussed in Section IV.2.8). The language specification formalizes the most common actions and targets relevant to cyber defense functions and defines command arguments for additional granularity. Furthermore, the specification elaborates on the available target and data types, includes serialization and encoding requirements, and presents examples of commands and responses in JSON. Examples of common actions and targets can be seen in Figures IV.5 and IV.6.

OpenC2 Actions (Defined in the Language Specification)	
Name	Description
allow	permit access or execution of a target
cancel	invalidate a previously issued action
contain	isolate a file, process, or entity so that it cannot modify or access assets or processes
delete	remove an entity (e.g., data, files, flows)
deny	prevent a certain event or action from completion, such as preventing a flow from reaching a destination or preventing access
detonate	execute and observe the behavior of a target (e.g., file, hyperlink) in an isolated environment
investigate	task the recipient to aggregate and report information as it pertains to a security event or incident
locate	find an object physically, logically, functionally, or by organization
query	initiate a request for information
restart	stop then start a system or an activity
scan	systematic examination of some aspect of the entity or its environment
start	initiate a process, application, system, or activity
stop	halt a system or end an activity

Figure IV.5: Subset of OpenC2 Actions Defined in the Language Specification

As observed in Figure IV.6, OpenC2 targets may be further refined by target type definitions that represent the combination of properties that make up a target, and specify their custom format attributes which are accurately described by authoritative resources, be they RFCs or other external specifications. For example, the target "file" is of target type "File". The target type "File" is

OpenC2 Targets (Defined in the Language Specification)		
Name	Type	Description
artifact	Artifact	an array of bytes representing a file-like object or a link to that object
command	String	a reference to a previously issued Command
device	Device	the properties of a hardware device
domain_name	Domain-Name	a network domain name
email_addr	Email-Addr	a single email address
features	Features	a set of items used with the query action to determine an actuator's capabilities
file	File	properties of a file
ipv4_connection	IPv4-Connection	a 5-tuple of source and destination IPv4 address ranges, source and destination ports, and protocol
ipv6_connection	IPv6-Connection	a 5-tuple of source and destination IPv6 address ranges, source and destination ports, and protocol
mac_addr	MAC-Addr	a Media Access Control (MAC) address - EUI-48 or EUI-64
process	Process	common properties of an instance of a computer program as executed on an operating system

Figure IV.6: Subset of OpenC2 Targets Defined in the Language Specification

the combination of "name", "path", and "hashes" properties, that are of types "String", "String", and "Hashes", respectively. The defined data type "Hashes" can be "md5", "sha1", or "sha256" and are expressed in binary or hexadecimal. Also they should be semantically validated based on their authoritative definition in their respective RFCs (RFC 1321 [12], RFC 6234 [5], and RFC 6234 [5]).

The language defines two payload structures: Command and Response. A command is an instruction from one system known as the producer to one or more systems known as the consumer(s) to act on the command's content. A command is comprised of four main components, of which two are required, and two are optional. The required components are the action-target pair, and the optional components are the command arguments and the actuator specifiers. Command arguments influence a command by providing additional information on how, when, and where should be performed. Moreover, command arguments can be used to convey the need for acknowledgment or additional status information about the execution of a command. Actuator specifiers further identify an actuator to some level of precision, such as a specific actuator or a group of actuators. A command can also contain an optional command identifier for tracking and referencing related commands and responses. The response is a message sent from the recipient of a command. Response messages provide acknowledgment, status, results of a query, or other requested information. At a minimum, a response will contain a status code to indicate the result of performing a command.

IV.2.7 OpenC2 Function-relevant Profiles

Actuator profiles allow cyber defense systems to interoperate in the context of particular cyber defense functions. A cyber defense system that serves as OpenC2 consumer can support one or more profiles based on its capability. For instance, a firewall is a policy enforcement mechanism that restricts or permits traffic based on static values such as source and destination address, protocol, and ports. A firewall as a cyber defense function can permit stateless or stateful packet filtering, and it can be dedicated (hardware-based) or integrated into other technologies and devices for adding a layer of security (software-based), such as networking and end devices. The engineering decision of defining profiles that are function-centric rather than device-centric is based on the observation that profiles for devices tend to overlap (key/value repetition), with at times only a few of their properties being different. For example, devices and technologies that support stateless and stateful packet filtering, such as network firewalls, host firewalls, cloud-based firewalls, and proxy firewalls, to name a few, share common characteristics in terms of operation, management, and configuration, meaning that an individual profile for each device would be very similar if not identical in some cases. However, all the aforementioned devices and technologies support stateful packet filtering, resulting in the creation of one common profile for use.

An actuator profile (specification) is created using a subset of the general language, and it may also extend the language by defining additional actions, targets, command arguments, and actuator specifiers relevant to a particular cyber defense function. In addition, an actuator profile includes conformance rules and requirements for seamless communication between producers and consumers. Creating function-relevant profiles (actuator profiles) is an iterative process that demands stakeholder, and specifically, vendor support. Security vendors and other organizations create custom actuator profiles and integrate OpenC2 into their cyber defense products and operations (based on use cases relevant to their organization or industry) for testing purposes, but also for supporting the development of OpenC2 by providing prototypes and reference implementations, where their cyber defense technologies can consume commands natively. The OpenC2 technical committee evaluates where function-relevant custom actuator profiles cluster to drive the development of new standard actuator profiles.

A consumer may support multiple actuator profiles. As mentioned in Section IV.2.3, a producer can request information about the versions of OpenC2 language, actuator profiles, and action-target pairs a consumer supports. It is possible for a consumer to support a limited range of action-target pairs of a profile but still claim conformance with the actuator profile. An example query for requesting the capability of a consumer can be seen right below.

```
{
   "action":"query",
   "target":{
      "features":[
         "versions",
         "profiles",
         "pairs"
      ]
   }
}
```

An example response from a consumer providing the versions of OpenC2 language, actuator profiles, and action-target pairs supported can be seen right below.

```
{
   "status":200,
   "results":{
      "versions":["1.0"],
      "profiles":["slpf-1.0"],
      "pairs":{
         "allow":["ipv6_net"],
         "deny":["ipv6_net"],
         "query":["features"],
         "delete":["slpf:rule_number"],
         "update":["file"]}
   }
}
```

Each OpenC2 specification, including actuator profiles, possesses a unique name in the form of a URI used to identify the document. A unique name ensures that all objects are identifiable and unambiguously referenced. For the shake of brevity in this paper, only the Stateless Packet Filtering actuator profile (SLPF) version 1.0 is presented [3].

Actuator Profile for Stateless Packet Filtering (SLPF): a stateless packet filter allows or blocks specific traffic based on a defined set of security rules and does not consider traffic patterns, connection state, data flows, applications, or payload information. The SLPF profile specifies the set of actions, targets, specifiers, and command arguments that are relevant to stateless packet filtering functionality. Through this command set, cyber security orchestrators may gain visibility into and provide control over the SLPF functionality in a manner that is independent (agnostic) of the instance (vendor-technology/device) of the SLPF function. A high-level overview of the SLPF actuator profile functionality is presented in Figure IV.7.

Actuator profiles define the language extensions that are meaningful and possibly unique to the actuator. For example, as seen in Figure IV.7, the SLPF

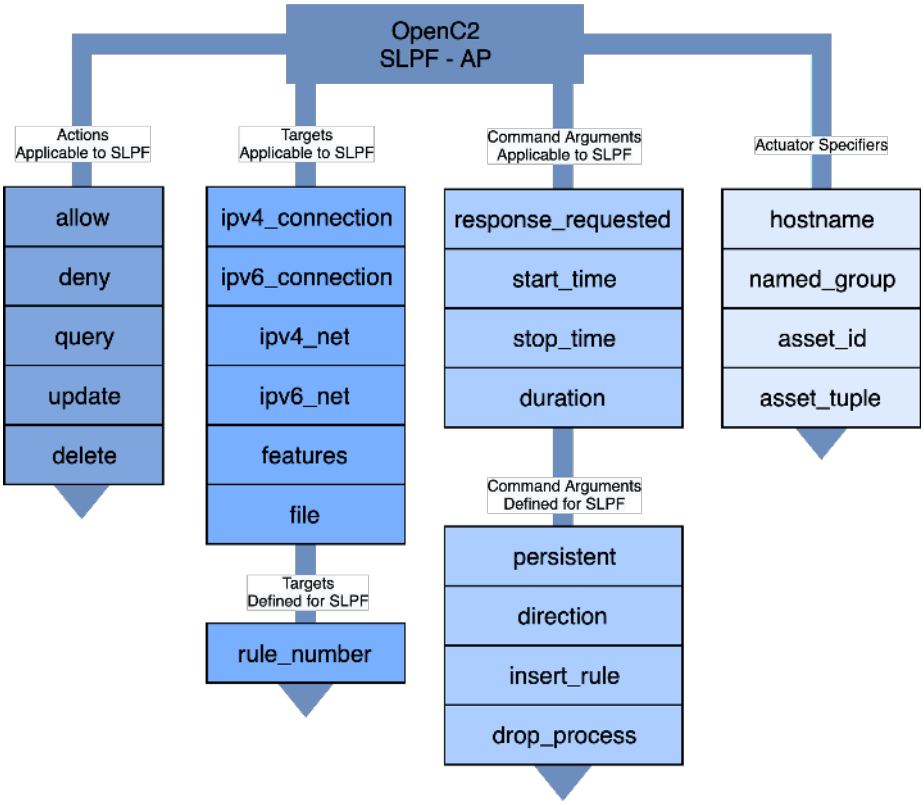

Figure IV.7: Overview of OpenC2 Stateless Packet Filtering Actuator Profile

profile defines multiple command arguments and one target introducing refined SLPF functionality in addition to the properties adopted from the language specification (OpenC2 properties from the general language specification that apply to SLPF). Figure IV.2 presents the action-target combinations that are syntactically valid when using the SLPF specification. Furthermore, Figure IV.8 elucidates the command arguments supported by each action.

Briefly, the additional SLPF-defined target "rule_number" combined with the action "delete" removes existing rules from a firewall rule-set. Also, the SLPF command argument "persistent" designates whether new firewall rules are persistent or not. The "direction" argument specifies whether firewall rules apply to incoming traffic, outgoing traffic, or both. The command argument "insert_rule" is used to specify a rule number to a new entry. The argument "drop_process" defines how the actuator handles denied packets. For example, an actuator can send a notification to the source of the packet or drop the traffic and send a false acknowledgment.

Command Arguments Matrix					
	Allow target	Deny target	Query features	Delete slpf/rule_number	Update file
response_requested	●	●	●	●	●
start_time	●	●		●	●
stop_time	●	●			
duration	●	●			
persistent	●	●			
direction	●	●			
insert_rule	●	●			
drop_process		●			

Figure IV.8: OpenC2 SLPF Command Arguments Matrix

IV.2.8 OpenC2 Transfer Specifications

A transfer specification defines how a particular transfer protocol is used for exchanging OpenC2 messages between producers and consumers. A transfer specification is agnostic of the content of a message, and similarly, the content of a message is agnostic of the underlying transfer protocol used. The language specification defines a set of message elements (Figure IV.9) that can be represented and used within the headers of a transfer protocol or some of them potentially within the body of OpenC2 messages to ensure interoperability and robust message exchange between technologies.

Common Message Elements		
Name	Type	Description
content		message body as specified by content_type and msg_type
content_type	String	media type that identifies the format of the content, including major version. For example, when using HTTPS: application/openc2-cmd+json;version=1.0
msg_type	Message-Type	the type of OpenC2 message (command/response)
status	Status-Code	populated with a numeric status code in responses
request_id	String	a unique identifier created by the producer and copied by the consumer into all responses, in order to support reference to a particular command, transaction, or event chain
created	Date-Time	creation date/time of the content
from	String	authenticated identifier of the creator of a message or authority for execution of a message
to	ArrayOf(String)	authenticated identifier(s) of the authorized recipient(s) of a message

Figure IV.9: Common Message Elements for Transfer Protocols

According to conformance clauses one and twelve in the language specification, OpenC2 producers and consumers should support one or more OpenC2 transfer specifications, which identify underlying transport protocols that can provide

authenticated, ordered, lossless delivery of uniquely identified OpenC2 messages. OpenC2 can be layered over any standard transfer and transport protocol. The transfer specifications utilize existing protocols and standards to implement OpenC2 in specific environments based on their requirements, capability, and constraints. An example is HTTP versus MQTT, where the first is recommended for fast and reliable networks, whereas the second is a better choice for networks that experience varying levels of latency due to occasional bandwidth constraints or unreliable connections. For the shake of brevity in this paper, only OpenC2 over HTTP(S) version 1.0 is presented [9].

OpenC2 Messages over HTTP(S): the Hypertext Transfer Protocol (HTTP) over Transport Layer Security (TLS) is one of the recommended transfer mechanisms for OpenC2 messages due to its broad availability and ability to transfer information in TCP/IP networks securely. OpenC2 over HTTP(S) is suitable for operational environments where connectivity is highly available and of sufficient bandwidth such that no appreciable message delays or dropped packets will be experienced.

The HTTP(S) Transfer Specification version 1.0 [9] provides guidance on how to utilize HTTP and the Transport Layer Security cryptographic protocol for exchanging OpenC2 messages. Each endpoint of an OpenC2 over HTTP interaction has both an OpenC2 role and an HTTP function. OpenC2 producers act as HTTP clients and transmit commands to consumers that act as HTTP listeners (server). A producer can issue OpenC2 commands using only the HTTP Post method, and a consumer can respond with an HTTP response message. As mentioned above, the OpenC2 language specification requires the use of two message elements for ensuring interoperability, namely "content_type" and "message_type". When OpenC2 command messages are sent over HTTP, the message type is "request" (HTTP request), and the content type follows the syntax [application/openc2-cmd+[serialization];version=[version]], where the media type, the serialization format, and the major version of the OpenC2 language utilized, are specified. This approach is built on the mechanics used to generate the related HTTP Content-Type header. For example, an OpenC2 command in JSON and conformant with the OpenC2 Language Specification version 1.0 and the Specification for Transfer of OpenC2 Messages via HTTP(S) version 1.0 should populate an HTTP Content-Type header "application/openc2-cmd+json;version=1.0". Equally, an OpenC2 response uses the message type "response" (HTTP response) and content type [application/openc2-rsp+[serialization];version=[version]].

Another considered approach that can provide a more transport-independent design to use the message elements over transfer protocols indistinctively is the inclusion of message elements into an OpenC2 construct that is part of the OpenC2 message body. This approach also has the benefit of being a better fit to transfer protocols that do not contain directly mappable headers or support custom ones.

According to the HTTP(S) Transfer Specification version 1.0, other header fields that SHOULD be populated when sending OpenC2 messages over HTTP are the "Cache-Control" that specifies what may be stored in caches on any

of the systems engaged at an OpenC2 transaction, the HTTP X-Request-ID that accommodates the "request_id" string (defined in the OpenC2 language specification, see Figure IV.9) populated by a producer, and "Date" that reflects the date and time at which the message was originated in the preferred IMF-fixdate format and conditions, as defined in Sections 7.1.1.1 and 7.1.1.2 of RFC 7231 [6]. Furthermore, the specification specifies other header fields that MUST be used when sending OpenC2 commands (HTTP request) over HTTP, such as the "Host" that specifies the hostname of the HTTP server and the listening port, the "Content-type" as described above, and the "Accept" header that advertises which content types, expressed as MIME types, the client is able to understand (e.g., "application/openc2-rsp+json;version=1.0").

The need for confidentiality, identification, and authentication when sending OpenC2 messages is addressed with the use of Transport Layer Security (TLS) cryptographic protocol. As stated in the HTTP(S) Transfer Specification version 1.0, OpenC2 endpoints must accept TLS version 1.2 connections or higher and must not support older TLS or Secure Sockets Layer (SSL) versions. The TLS sessions must not use NULL cipher suites because they do not provide confidentiality for the TLS traffic, and OpenC2 endpoints supporting TLS version 1.2 must not use any of the blacklisted cipher suites identified in Appendix A of RFC 7540 [1]. Finally, OpenC2 endpoints supporting TLS version 1.3 must not implement zero round trip time resumption (0-RTT) as it has been proved to be prone to replay attacks.

IV.3 OpenC2 Use Case Implementation and Results

The use case presented in this section is a consolidated effort by the University of Oslo, AT&T, the University of North Carolina, and the Cyber Defense Institute of Japan to demonstrate interoperable tactical responses in cyber relevant time using OpenC2. We present a proof-of-concept implementation of OpenC2 across several systems and components that support stateless packet filtering. The four prototypes developed conform with the Stateless Packet Filtering actuator profile version 1.0 and are all interfaces for different products that serve the same purpose.

The implementation comprises four integrated prototypes over two transfer protocols. In particular, the University of Oslo engineered an adapter that can utilize OpenC2 commands for device management and configuring firewall rules on Cisco routers that support access control lists. AT&T presented an adapter for configuring the packet filters of Amazon, Google, and Microsoft cloud platforms. The University of North Carolina developed an interface for configuring Linux iptables with OpenC2 commands. iptables is a command-line packet filtering utility that uses policy chains to allow or block traffic using the Netfilter framework provided by the Linux kernel. The Cyber Defense Institute of Japan created an OpenC2 interface for configuring firewalld. firewalld is a firewall tool for Linux operating systems. Like iptables, firewalld uses the kernel's Netfilter framework for filtering traffic. Figure IV.10 presents a high-level

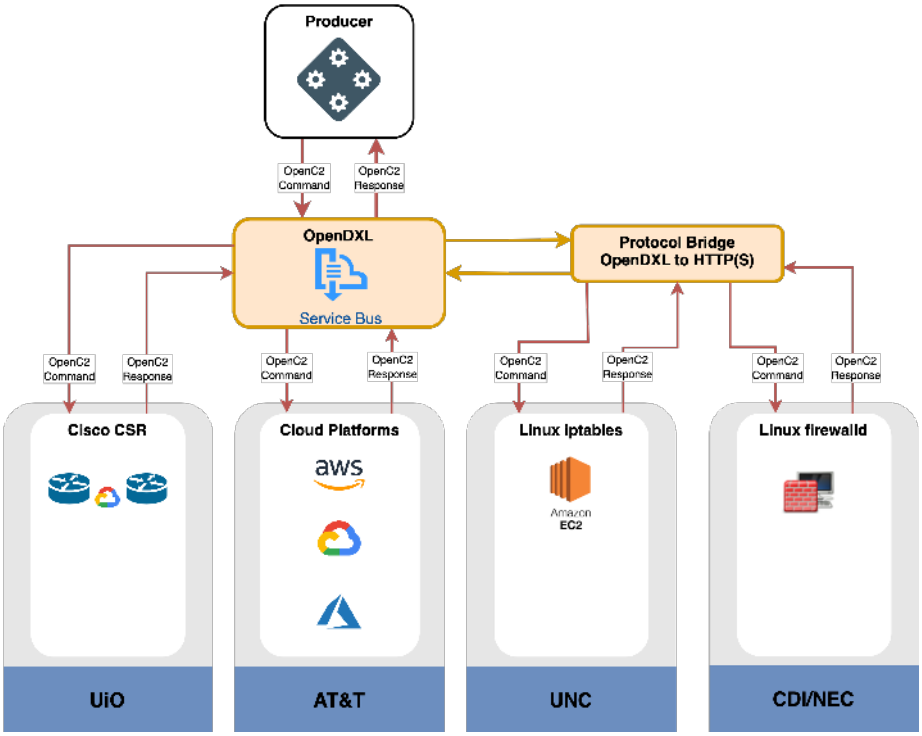

Figure IV.10: Prototype Integration over DXL and HTTPS

diagram of the integration of the prototypes and the communication of OpenC2 messages.

The implementations communicated OpenC2 messages over the DXL (OpenDXL) message fabric and HTTPS. OpenDXL or Open Data Exchange Layer is a publish-subscribe message fabric that allows applications to publish and subscribe to message topics. An OpenC2 producer can publish an OpenC2 command on a relevant topic such as "network/packet-filters", and the producers can subscribe on the same topic to receive the published messages. A protocol bridge was created to facilitate message exchange for the implementations that were using an HTTP interface (listener) to communicate. The integration and message exchange would be more straightforward if an orchestrator has been used for registering devices, and for certificate-based authentication management.

The experiments demonstrated and proved that OpenC2 enables command and control of cyber defense systems and components at machine speed and in a manner that is agnostic of the underlying technologies and protocols utilized. The integration confirmed that all prototypes could receive a single command (e.g., deny/block a particular IP address) and act on that command in the same way, proving that the language, the transfer specifications, and in particular the actuator profiles are robust enough to be used unambiguously among a set

of different cyber defense systems and components of the same cyber defense function.

IV.4 Conclusion

OpenC2 enables command and control of cyber defense components and systems in cyber-relevant time and in a manner that is agnostic of any of the underlying products, technologies, transfer mechanisms, or other implementation aspects. This is achieved by introducing a common language that allows seamless integration and enables interoperability between cyber defense technologies. OpenC2 cyber defense function-specific profiles (actuator profiles) can be integrated into cyber defense components and systems for introducing native OpenC2 interfaces, shifting traditional command and control approaches that are based on proprietary APIs (where a set of requirements that govern how one application can communicate and interact with another is provided), to an approach that response actions can be governed by one common language and be vendor agnostic. A language such as OpenC2 is necessary but insufficient to enable coordinated cyber responses that occur within cyber-relevant time. Other aspects of coordinated cyber response such as sensing, analytics, and selecting appropriate courses of action should be provisioned. OpenC2 will provide the capability of executing the chosen actions.

Acknowledgements. This research was supported by the research projects CyberHunt (Grant No. 303585) and ACT (Grant No. 256785) funded by the Research Council of Norway.

OpenC2 is a community-driven effort within OASIS. The specifications analyzed in this research paper are the result of the work and contributions of multiple member organizations. A list of participants involved in this effort is available on the official OASIS OpenC2 web page[2] and at the end of each specification annexed. Furthermore, the authors would like to thank Mr. David Lemire (Huntington Ingalls Industries) for the valuable suggestions that helped improve the manuscript.

References

[1] Belshe, M., Peon, R., and Thomson, M. *Hypertext Transfer Protocol Version 2 (HTTP/2).* RFC 7540. [Online]. Available: http://www.rfc-editor.org/rfc/rfc7540.txt. RFC Editor, May 2015.

[2] Bray, T. *The I-JSON Message Format.* RFC 7493. [Online]. Available: https://tools.ietf.org/html/rfc7493. RFC Editor, Mar. 2015.

[3] Brule, J., Sparrell, D., and Everett, D., eds. *Open Command and Control (OpenC2) Profile for Stateless Packet Filtering Version 1.0.* OASIS Committee Specification 01. 2019.

[2] https://www.oasis-open.org/committees/openc2

[4] Crawford, M. *Binary Labels in the Domain Name System*. RFC 2673. [Online]. Available: https://tools.ietf.org/html/rfc2673. RFC Editor, Aug. 1999.

[5] Eastlake, D. and Hansen, T. *US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)*. RFC 6234. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6234.txt. RFC Editor, May 2011.

[6] Fielding, R. and Reschke, J. *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. RFC 7231. [Online]. Available: http://www.rfc-editor.org/rfc/rfc7231.txt. RFC Editor, June 2014.

[7] Herring, M. J. and Willett, K. D. "Active Cyber Defense: A Vision for Real-Time Cyber Defense". In: *Journal of Information Warfare* vol. 13, no. 2 (2014), pp. 46–55.

[8] IEEE Standards Association. *Guidelines for Use of Extended Unique Identifier (EUI), Organizationally Unique Identifier (OUI), and Company ID (CID)*. 2017.

[9] Lemire, D., ed. *Specification for Transfer of OpenC2 Messages via HTTPS Version 1.0*. OASIS Committee Specification 01. 2019.

[10] Mavroeidis, V. and Bromander, S. "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence". In: *Proceedings of the 2017 European Intelligence and Security Informatics Conference (EISIC)*. IEEE. 2017, pp. 91–98.

[11] OASIS. *OASIS Open Command and Control (OpenC2) TC*. [Online]. Available: https://www.oasis-open.org/committees/openc2/.

[12] Rivest, R. L. *The MD5 Message-Digest Algorithm*. RFC 1321. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1321.txt. RFC Editor, Apr. 1992.

[13] Romano, J., and Sparrell, D., eds. *Open Command and Control (OpenC2) Language Specification Version 1.0*. OASIS Committee Specification 01. 2019.

[14] Willett, K. D. "Integrated Adaptive Cyberspace Defense: Secure Orchestration". In: *Proceedings of the International Command and Control Research Technology Symposium (ICCRTS), Annapolis, MD*. 2015.