Enabling New Technologies for Cyber Security Defense with the ICAS Cyber Security Ontology

Malek Ben Salem
Accenture Technology Labs
Arlington, VA
malek.ben.salem@accenture.com

Chris Wacek
Invincea Labs
Arlington, VA
christopher.wacek@invincea.com

***Abstract*—Incident response teams that are charged with breach discovery and containment face several challenges, the most important of which is access to pertinent data. Our TAPIO (Targeted Attack Premonition using Integrated Operational data) tool is designed to solve this problem by automatically extracting data from across the enterprise into a fully linked semantic graph and making it accessible in real time. Automated data translation reduces the costs to deploy and extend the system, while presenting data as a linked graph gives analysts a powerful tool for rapidly exploring the causes and effects of a particular event. At the heart of this tool is a cyber security ontology that is specially constructed to enable the TAPIO tool to automatically ingest data from a wide range of data sources, and which provides semantic relationships across the landscape of an enterprise network. In this paper we present this ontology, describe some of the decisions made during its development, and outline how it enables automated mapping technologies of the TAPIO system.**

***Index Terms*—cyber security, ontology, cyber analysis, semantic technologies, ontology patterns, forensic analysis**

I. Introduction

Cyber security suffers from a critical talent shortage [10]. According to Cisco's 2014 Annual Security Report, over a million cyber security positions went unfilled worldwide [7], and companies are constantly facing challenges recruiting and retaining skilled security professionals. As targeted attacks on computer networks have become more prevalent, the need for tools that allow analysts to effectively hunt for malicious activity has exploded. The existing solutions in this space - security event information management (SIEM) systems - seek to ingest and aggregate all of the logs or events that might be of interest to a defender, generally storing each according to its source schema. Adding new data sources to these systems requires engineering new "connector" to be able to source and parse the data. Tracking the broader context is left to the skills and experience of the analyst, and correlation between events is frequently a manual and mental process [14].

Our prototype TAPIO system seeks to upend this paradigm by ingesting events and the context around them, and leveraging our cyber security ontology as a guide for transforming that information into a fully linked semantic graph. By producing a fully linked semantic graph, we change the task of correlating and connecting events and actions into a simple traversal of the data graph. The aim is to reduce the specialized security knowledge required for an analyst to be effective at understanding and evaluating the threat presented by a given alert, and to enable them to hunt for threats intuitively.

To enable this, we leverage semantic web technologies such as Resource Description Framework (RDF) and Web Ontology Language (OWL). TAPIO agents running locally on devices across the network 1) automatically interpret the data sources and translate the unknown schemata of each source to a common ontology, 2) link the data, and 3) store it in a local RDF store. The information is then accessible through a real-time query capability to security analysts. At the heart of the capabilities listed above lies the cyber security ontology, which we refer to as the Integrated Cyber Analysis System (ICAS) ontology.

Typically, the primary goal of any ontology is to enable knowledge sharing and re-use. An ontology can be considered as a conceptualizing of a certain specified domain at some level of abstraction. By creating a common lexicon around *what exists* in the domain, an ontology enables the creation of tools and components that are capable of storing and exchanging information about that domain. There are many existing abstractions of computer and network operations - one need only lightly review the veritable alphabet soup of ontologies and taxonomies out there to make that clear [15, 25, 23, 24, 22] .

However, in the case of TAPIO the ontology has a second purpose - to support automated data translation by the system, and in doing so lower the barriers to providing access to new data sources to security analysts. This dual purpose led us to construct the ICAS ontology presented in this paper, building upon the great body of work in knowledge representation that has been developed in previous ontologies and taxonomies. In this paper we:

- present the ICAS cyber security ontology that models security related elements and the semantic relationships between them;
- identify several necessary considerations when developing an ontology with the dual goal of providing a data schema and enabling automated translation to it; and
- provide an overview of how the TAPIO system uses the ontology to support its data mapping.

The remainder of this paper is organized as follows. In Section II, we review related work. We then present the main ontology requirements in Section III. We present the

ontology and the design considerations that were involved in constructing it in Section IV. In Section V, we outline how the ontology enables the cyber defense TAPIO tool. We focus in particular on how the ontology enables the extraction of data schemata from previously unseen data, thus allowing the interpretation of that data and its mapping into a graph database. We discuss some of the challenges and limitations of using and maintaining the ontology in practice as well as some directions of our future work in Section VI.

II. Related Work

In this section, we review prior work related to developing our cyber security ontology. We start by covering previous general cyber security ontologies and security standards relevant to our work. Then we review some event models that we leveraged in this work.

A. Security Standards

The US government launched the Information Security Automation Program (ISAP) to develop security standards in order to enable the automation of technical security operations. Several standards have been developed including the Common Vulnerabilities and Exposures (CVE), Common Configuration Enumeration (CCE), Common Weakness Enumeration (CWE), Common Vulnerability Scoring System (CVSS), the Extensible Configuration Checklist Description Format (XCCDF), the Open Vulnerability and Assessment Language (OVAL), and the Common Attack Pattern Enumeration and Configuration (CAPEC). These standards are for the most part taxonomies, captured in XML format. Therefore, they fail to capture the semantics and relationships connecting them, thus making them not very useful for automated vulnerability management. To overcome this shortcoming, Wang and Gui leveraged CVE, CWE, CVSS and CAPEC to build an ontology for vulnerability management (OVM)[27]. The ontology allows the user to compare similar products based on their vulnerability information.

B. General Security Ontologies

There have been a few early attempts to build a cyber security ontology.

A few general security ontologies have been proposed. Herzog's Ontology [9] had 463 security-related concepts, and Fenz's ontology [8] had about 635 security-related concepts. The latter focused on threats targeting hardware assets rather than information assets. The Navy Research Lab's security ontology [11] had only 75 security concepts. A more detailed comparison of these ontolgies is available in [20].

The main weakness of these ontologies is that they focus more on objects rather than on events. Capturing events is needed to reconstruct the timeline of a cyber attack. The existing general cyber security ontologies – while they do cover many of the cyber security-related concepts – fail to provide a framework for capturing cyber and security events and the relationships between them. However, exploring the connections between events is important to improve situational awareness during and after a cyber attack, and is therefore paramount to any digital forensic investigation or tactical cyber defense.

Obrst et al. presented an overall architecture for a cyber security ontology that focused on malware in their seminal "Developing an Ontology for the Cyber Security Domain" paper [16]. The architecture included the foundational ontologies, which they referred to as the upper ontologies. These constitute the domain-independent ontologies describing concepts such as *entities* and *collections*. Mid-level ontologies in their architecture are ontologies that make assertions that span multiple domains, such as OWL-Time. At the lowest level of the architecture are the domain ontologies which represent domain-specific concepts. The ontology described in this work follows the same tiered architecture.

Another cyber security ontology was developed by Oltramari et al. [17]. The ontology, which is still a work in progress, aims to cover operational cyber defense by addressing the human factor besides the technological spectrum in complex cyber defence **operations**. The ontology focuses on cyber defender operations, and is complementary to our ontology, which focuses on attack discovery and contextualization.

C. Event Modeling

In [21], Scherp et al. presented a formal representation of events that enables capturing and representing human experience when dealing with various situations outside the cyber context. They developed several ontology design patterns for event correlation, causation and documentation. The latter is used to capture additional documentation entered by humans as they intervene, experience and interpret while being involved in these events.

Recently, Chabot et al. proposed an approach for reconstructing the timeline of events for digital forensic investigations [5, 6]. They developed an event model designed specifically for digital forensic investigations by linking events to their digital footprints. For example, an authentication event is linked to the authentication log entry, which constitutes the footprint of the event. This event model allows not only for the temporal correlation of events, but also for event subject and object correlation, as well as rule-based correlations. The latter can capture cause-effect relationships between events that can be predefined by subject matter experts.

There is a trade-off between capturing more semantics about events to provide richer event contextualization and the size of the triple store where this information is stored. In this work, we opted for the more light-weight event model described in [21], applied to the cyber environment, in order to limit the size of the triple stores at the endpoint devices and therefore speed up running queries against these stores. This tradeoff becomes even more important for sensors and smaller devices with limited memory and computational power, as we deploy our integrated cyber analysis systems on IoT (Internet of Things) networks and push more cyber analysis to the edge of the network.

III. Ontology Requirements

In this section, we introduce the cyber security ontology and start by listing the key ontology requirements elicited during the ontology design process:

- Representation of any entity (physical/cyber) that needs to be deployed and managed in an enterprise's IT network: One of the main goals of our ontology is to map a complete picture of the IT environment for the security analyst and to provide general coverage of the cyber security and digital forensics domains. The ontology should encompass all such entities that would be involved in an incident response action or digital forensic investigation. There are well defined ontologies in various domains, which we re-use and integrate into the ICAS ontology. These include the Ontology for Vulnerability Management described above [27]. We have also developed an extensive set of domain ontologies, including the memory artifacts ontology and the registry ontology.
- Representation of lower level events: Beyond capturing the physical and cyber entities, the ontology should capture cyber security events and digital forensics/evidence such as login sessions, software installations, etc.
- Temporal and local contextualization of events: The significance of cyber events depends on their local and temporal context. The chronological order of (security) events may suggest underlying connections between them, such as two consecutive authentications from remote locations that are geographically widely separated occurring within a short period of time. So to correlate these events and draw connections between them, the ontology must support comparing their temporal dimensions.
- Abstraction of higher level events: Cyber events are interdependent and could be correlated/aggregated to higher-level events; For instance, several port scanning events could be abstracted into the reconnaissance step of an attack.
- Abstraction of cyber security attacks and attacker's TTPs (Techniques, Tactics and Procedures): Attack progression interpretation requires reconstructing the timeline of the events relevant to the attack, and reasoning on temporal events. It also capturing general concepts related to security events, such as attack patterns.
- Annotation of extracted facts (relations between entities) with confidence values that reflect the trust in the correctness of the statement: Uncertainty relates to events based on missing or partial sources, which can lead to dealing with unknown information or contradictory facts, such as the example of the two authentication events from two different places within a short window of time.
- Removal of wrong/invalidated statements in order to obtain a high quality knowledge base: Events and entities need to be annotated with validity times and confidence scores, so that (temporal) consistency constraints may be checked and inconsistencies eliminated from the knowledge base. A high-quality knowledge base is needed for the reasoning engine.

IV. ICAS Ontology Design and Development

Our ontology is based on the foundational ontology DOLCE + DnS Ultralite (DUL) [19]. DUL is a lightweight version of the DOLCE [1] + DnS [18] ontology. It provides the upper level concepts that form the basis of interoperability between lower-level ontologies. So we adopted DUL's design decision which distinguishes between *events* and *objects*, where *events* unfold over time and *objects* unfold over space.

One way of thinking about this is that *objects* are elements that exist through time - the same process can be observed at many different instants in time - while *events* relate those *objects* at a specific moment in time.

The ICAS ontology is organized as a collection of sub-ontologies that capture specific conceptual areas. At present there are 30 sub-ontologies ranging from the highly specific *Memory Artifacts* which models the types of elements that might be found in a memory dump, to the much more general *Process* ontology which describes general information about running processes. The sub-ontologies are generally an organizational construct - object properties create regular connections between them.

In keeping with the intention to model all security relevant elements of an enterprise network, the ICAS ontology models network concepts in addition to the host-level observables described above, although networking information is much more frequently expressed via *events* than via *objects*. It does not attempt to replicate network traffic information at the most granular level - a task better suited to formats like NetFlow.

There are many ways to represent concepts as *objects* within an ontology, and the representation used really depends on the intended use of the ontology. For instance, a file can be represented as a single entity with a path, size, and a hash value. Alternately, an ontology can mirror the file system model where file contents are represented distinctly from its location in the file system.

In developing the *object* representations in the ICAS ontology we followed the following guidelines:

- represent each concept as close to reality as possible, without requiring platform specific distinctions
- represent the state of the world, not the rules of the world

In the case of files and file paths the first guideline led us to represent them separately, since this makes it possible to record observations of file paths without requiring knowledge of the underlying file, and permits the contents of a file to be linked from multiple different file paths (as is the case in the real world).

Windows Registry keys exemplify the other side of the first guideline. Registry Keys contain configuration information for Windows, but other platforms don't use a registry in the same way. However, all platforms have some variation on a key-value configuration store accessible by paths, whether registry paths or file system paths. As such, the ICAS ontology defines a generic `ConfigurationFileName` object that is used to represent Windows Registry keys, OS X

`.plist` files, and Linux `/etc` files. Values keys within the registry or within the configuration files can be represented with `ConfigurationKey` objects attached to the `ConfigurationFileName` object.

The second guideline has two goals: increased flexibility and reduced complexity. Consider the representation of file paths in a file system. One could model the directory structure as a complete sequence of related objects (i.e. `C:\Windows\System32` is three `Directory` objects related via the `contains` relationship, and the entire file system could be walked in the same way that one can traverse a directory tree. Alternately, this could be represented as a single `Directory` object with the entire path stored as a Datatype Property.

We choose the latter method to represent file system objects (and other concepts with hierarchical relationships such as IP networks), for several reasons. The first is that it vastly reduces the complexity of object representations, which in turn reduces the complexity of data translation and data querying. When translating data into the ontology, the former method requires a specialized processor to recognize file paths and translate them into a sequence of hierarchically related objects; similarly when composing queries, literal file paths must be decomposed into a sequence of directory objects for the query engine. The translations on each side are specialized and platform specific, which introduces opportunities for error.

The other reason is that the ICAS ontology is intended to support a tool to represent data for analysts, and human analysts are very capable of applying their own understanding to the data presented to them. For instance, that shared prefixes in file paths indicate shared hierarchical structure is commonly known. A search for all files in a directory can easily be composed by applying the analysts own understanding using a substring filter. This means that the analyst can apply their intuition easily and is not forced into understanding a rigid structure that might differ from their interpretation.

Combined, these two guidelines provide a flexible, powerful framework for object representation in the ontology that enables describing entities observed in the world, without attempting to describe the specifics of all the protocols within that world.

To make the cyber security ontology enable attack discovery and security analysis, it was critical to use a robust methodology for representing *events* and the relationships between them. Exploring connections between events is important to any cyber investigation. This requires knowledge about what events preceded, co-occurred with or succeeded others. It may require information about the time span of each event and which events occurred in the same time window. We model events as instants with a corresponding duration in time according to the W3C OWL Time ontology, which enables considering them as a sequence within a `Timeline` according to the model proposed in [26].

V. Applying the Ontology

The ICAS ontology described to this point is designed to provide an abstraction of the operation of a computer network, straddling the traditional demarcation lines between the host and network-attached devices with a focus on how elements inter-relate.

When cyber security analysts get access to the semantic relationships between information, this transforms the investigative task by natively supporting the natural human inclination to connect information. Instead of having to mentally track potentially relevant pieces of information and identify relationships on the fly, their mental resources are freed up to focus on the actual task: determining whether or not a pattern actually looks representative of malicious behavior.

A. TAPIO Overview

Our prototype TAPIO system is designed to enable this transformation by performing *ontology-guided data translation*. We describe this as *ontology-guided* because none of the techniques employed by TAPIO are specific to the cyber security domain; instead it draws information used for mapping directly from what is encoded in the ontology itself. An ontology for a different domain would enable TAPIO to translate data sourced for that domain.

In the cyber security setting, TAPIO searches for information on target systems and translates that information into linked data represented according to the ontology, without the use of handwritten rules or parsers. In this it effectively operates similarly to a database "view", enabling analysts to query linked data from across the network without knowledge of the underlying source. TAPIO consists of several components: native data sources, a data translation layer, RDF base storage, and a SPARQL query layer. The data sources are capable of retrieving data from a variety of locations, including host-based commands and APIs, remote HTTP and SQL connections, and network observation. The data translation layer accepts records from the data sources and converts them into RDF triples which are stored by the storage and the query layers.

TAPIO uses the ICAS ontology as both data schema and as an information source for automated data translation. The system stores all data in RDF triples that represent objects and properties as described in the ontology. This enables us to use the powerful capabilities of the SPARQL query language natively, where the ontology doubles as the definition of the schema against which queries can be executed.

The TAPIO system also provides feedback to the query interface about what data has been *realized*. That is, while the ICAS ontology describes the set of information that can be searched, only a subset of those objects and properties are likely to have been observed. We describe an observed ontology property or object as being *realized*. This feature means that analysts do not lose time by searching for data that will never be translated.

B. The Schema Translation Problem

One of the biggest issues for computer systems designed to provide data querying capabilities is that any data must be normalized and translated to fit the schema in use. This is true whether one is talking about a relational database schema or a full RDFS/OWL ontology such as the one used by TAPIO, and the growth in popularity of data federation and data warehousing has resulted in a variety of research efforts in this area [13, 3, 4, 2]. Much of the research focuses on translating schemata from one representation to another, e.g., mapping an XML representation of a running process into a relational database schema for that running process.

While we leverage many of the intuitions and suggestions proposed in this body of research, we suggest that the problem TAPIO seeks to address is somewhat distinct in that we seek to find a mapping from multiple *unknown* schemas to our ICAS ontology. Essentially we attempt to concurrently perform schema detection and schema translation by taking data from a source (which has a schema although we do not know what it is) and mapping it into subgraphs that are valid according to the ontology.

1) Assumptions and Resources: We make a number of reasonable assumptions about the data that we seek to translate:

1. Input data consists of a stream of *records*, where each record contains an arbitrary number of key-value pairs.
2. Key-value pairs that occur in a record together are related.
3. Keys are textual and convey some meaning related to the semantics of the data (i.e. they are not totally random).

At first blush, the first assumption does not seem reasonable; data is rarely structured in key-value form. However, we argue that almost any input data source, whether structured or unstructured can be transformed into sets of key-value pairs through techniques such as structure detection or named-entity recognition labeling. Fig. 1 shows how this might work for a common authentication log file. The TAPIO system attempts a variety of pre-processing steps to attempt to determine this structure. In the case of an authentication log containing lines like the one shown in Fig. 1, these techniques include a two pass-approach that searches for different *formats* within the log file using an unsupervised n-gram based clustering approach, and then attempts to identify *fields* within each format by observing which segments appear to be constants and which have high variance within the cluster. The details of this and the other techniques are not included here for space reasons.

The second assumption is reasonable so long as the source record is well-formed; it would be an exceedingly strange data source in which data contained within a given record was unrelated. This assumption is key to our ability to perform multi-origin mapping; we use it as a constraint on the candidate mappings we select by arguing that if candidate mappings cannot be easily related, then they are not likely to be correct because the underlying data is not related. At the same time, the processes which extract structured data from each data source are themselves heuristic and could inadvertently identify key-value pairs that do not represent real data; for instance, the word "user" in Fig. 1 is not a data field. To accomodate this reality, the TAPIO system assumes that keys within a record are related, but does not require that all identified keys are *used* in a mapping.

The third assumption is not a strict requirement, so long the same data field uses the same key over a sequence of input records. However, several of the techniques discussed in Section V-B2 operate on natural language and thus are improved if the assumption holds.

Under these conditions we have a couple of pieces of information that our mapping process can leverage: individual keys and values whether separate or combined, the set of keys in a record, the set of values for a given key over a sequence of records, or a combination of any of the above. Some of these features, such as the keys, incorporate semantic meaning, while others contain hints about the type of data that is embedded within them. A timestamp, for instance, has a reasonably distinctive format. In addition to the data itself, each record is annotated with some information about the source from which it was obtained or observed.

2) Mapping Pipeline: To maximize the chance of an accurate translation from key-value records to a linked subgraph, we perform the translation in several stages designed to iteratively identify candidate data translations, refine those candidates, and construct graph mappings using those candidates.

TAPIO's mapping pipeline is *data-oriented*: we restrict the set of potential candidates to the *DatatypeProperties* contained in the ontology, requiring that literals in the incoming records be mapped to data in the resulting graph. Objects in the resulting graph are then realized by implication, e.g. we have observed a `process#PID` property and its domain is `Process` thus implying that an object of that type exists.

The process of identifying candidate data translations is called *name resolution*. At this stage TAPIO uses a variety of algorithms in parallel, drawing techniques from machine learning, information retrieval, and other domains. Each *name resolver* produces a set of translations from the keys contained in the input data record to the candidates in the ontology. These candidates are then refined in a *meta name resolver*, whose role is to learn the efficacy of each *name resolver* algorithm in the first and use that to weight the results.

Given a refined set of candidates, we attempt to identify the best method for constructing a linked semantic subgraph from some subset of them. This *linking* stage applies an algorithm based on the method employed by Knoblock, et. al. [12] to each possible combination of the data candidates. The ontology is the source of information about how the semantic subgraph can be constructed. Each candidate datatype property causes the creation of node with of the type of its `rdfs:domain`. Multiple properties with the same domain are attached to the same object, unless they are duplicative with another property; in that case a second object is created. The linking algorithm then attempts to connect the objects by traversing *ObjectProperties* identified in the ontology. The

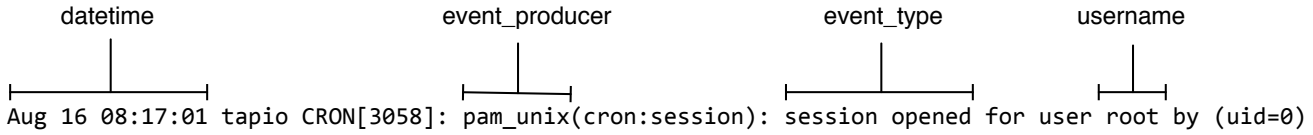

Fig. 1. An example of how unstructured data can be transformed to key-value pairs. Sample keys are shown above segments of the data

result is a connected subgraph generated based on the input combination of datatype properties.

The best resulting subgraph is selected using a scoring method that trades off between including additional candidates and the size of the resulting subgraph.

VI. Discussion

In the previous sections we have discussed our approach to constructing an ontology for representing all information that might be useful for a cyber defender, several elements that make building an ontology designed to be used by an automated mapping system difficult, and an overview of our prototype that attempts to perform that automated mapping. We now turn to a discussion of some of the practical challenges that our TAPIO system is likely to encounter in real-world operations.

A. Using Ontologies for Automated Mapping

RDFS/OWL ontologies provide a very flexible method for capturing and exchanging the semantics of data, but the very nature of linked data can present some challenges when attempting to automatically translate observed data into an ontological representation thereof. Semantic graphs encode data not only in *DatatypeProperty* edges (effectively data fields), but also in the *Class* of nodes and *ObjectProperties* connecting them. A correct translation from observed data to an ontology subgraph requires not only that the data fields are correctly aligned to datatype properties, but that the classes and semantic relationships between them are chosen correctly. This has several ramifications when designing an ontology to be used for automated mapping:

1) Data Domain Specificity: The first is that in any ontology used for automated mapping, domain and range labels become very important for both data and object properties. Our mapping algorithms attempt to map observed data to *DatatypeProperties*, i.e. to literal data labels. According to the definitions of RDFS, the existence of a property with a domain D allows us to infer that an object which is a member of the class D exists. We use this logic to instantiate an object of class D based on the existence of the datatype property. If we are not specific about domain values, whether through non-specification or overly generic specification, the system cannot infer the existence of specific objects. Consider the example ontology shown in Listing 1, which represents two types of objects - Users and Processes - with a single generic `name` data property. We call this property *generic* because the domain is the generic `owl:Thing` class instead of either `Process` or `User`. If the mapping algorithms correctly identify that an element of observed data maps to the `name` property, they still have little ability to guess which class it should be associated with.

```
:Process a owl:Class ;
    rdfs:label "Process"@en ;
    rdfs:subClassOf owl:Thing .

:User a owl:Class ;
    rdfs:label "User account"@en ;
    rdfs:subClassOf owl:Thing .

:name
    a owl:DatatypeProperty ,
      owl:FunctionalProperty ;
    rdfs:domain Owl:Thing ;
    rdfs:range xsd:string .
```

Listing 1. Example Ontology for User and Process with generic properties

In developing the ontology, we aimed to be as specific as possible with respect to datatype property domains. This means that the ontology often contains similar properties - `hasName` exists in both the `user` and `process` sub-ontologies - that at first glance seem like they should be consolidated into one generic property. However, each has a different dICAS omain so that we can assert the existence of a `UserAccount` or `Process` object respectively.

2) Ambiguity in Object Properties: As discussed, the automated mapper constructs a set of objects by inferring their existence after mapping literal data elements to datatype properties. At this point, each object is independent and the system still needs to identify the correct *ObjectProperties* to provide the semantic relationships between them.

It is quite natural for a human to consider multiple nuanced semantic relationships between objects and discern the correct one in any given situation. Consider the existence of a `Process` object and an `IPCSocket` object. There are several different possible relationships between these two objects - the process either `connectsTo` the socket, or it `binds` the socket. To a human looking at the input data, it is relatively easy to discern which of these two situations they are observing; but machines have far less perspective that they can bring to bear.

An automated mapping system has little option but to choose one (or both) of the properties at random. Leveraging OWL constructs such as `disjointWith` can avoid the latter case, but does little to alleviate the larger issue.

In the ICAS ontology, we take the approach of requiring that all semantically-related but specific *Object Properties* share a base property that the mapper can *fall back* to in the absence of more specific information. In the example outlined above, this means that we create a new property `usesSocket` and designate both `connectsTo` and `binds` as subproperties of this object property. We then permit the mapping algorithms to choose either the base property or at most one of its subproperties, giving them flexibility to be specific if there is supporting evidence, but allowing a generic fallback that does not imply semantics that are incorrect. At present, we permit only one tier of subproperties for simplicity.

B. Operating in a Dynamic World

Another challenge arises due to our use of the ontology as a *commitment* to a data representation in a distributed data storage system. As data is observed and collected, it is stored according to the semantics of the ontology at that point in time. In the event that the ontology needs to be updated to account for new information or new understanding, any existing data needs to be adjusted to avoid semantic *skew* between the incoming data and query layer and the existing data.

The naive solution to this problem is to simply discard historical data when the ontology is updated, on the assumption that changes to the ontology as a semantic model are rare events. This is an unsatisfying solution. We envision several more robust solutions to this problem. The first is to identify the conflicts between the existing and new ontologies and request that the user provide translation rules for how to update the existing data. The second solution is to tag all data with the version of the ontology under which it was mapped, and constrain searches to a particular semantic data slice. This latter method is easy to implement, but has the effect of partitioning the database and significantly reducing the efficacy of the tool. The former method of using translation rules to *migrate* the old data to the new schema maintains the utility of the system, but is much more expensive from the perspective of both knowledge engineering and system performance. We intend to explore both methods as TAPIO matures.

In our future work, we will also continue focusing on optimizing the ontology in order to improve query performance. We will explore the use of additional semantics to process complex events and achieve a better understanding of the relationships between them.

VII. Conclusion

In this paper we have suggested that semantic web ontologies can play an active role in developing new paradigms for computer security. In support of that argument, we describe our prototype TAPIO tool that uses the newly developed ICAS cyber security ontology to enable both description of semantically-related security data and machine-assisted translation of arbitrary sources to the ontology. We discuss some of the requirements and challenges presented by using semantic web technology in this dual role. Our goal is that TAPIO will help change the way we think about computer network defense by enabling analysts to seamlessly and intuitively hunt for targeted attacks, and that in doing so show the value of semantic ontologies in enabling new technologies for cyber security.

VIII. Acknowledgments

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA) Integrated Cyber Analysis System (ICAS) program. The views, opinions, and/or findings contained in this article are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] *A Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)*. URL: http://www.loa.istc.cnr.it/old/Papers/DOLCE2.1-FOL.pdf (visited on 06/08/2015).

[2] Paolo Atzeni, Paolo Cappellari, and Giorgio Gianforme. "MIDST: model independent schema and data translation". In: *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM. 2007, pp. 1134–1136.

[3] Paolo Atzeni, Paolo Cappellari, Philip Bernstein, et al. "Modelgen: Model independent schema translation". In: *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*. IEEE. 2005, pp. 1111–1112.

[4] Philip A Bernstein, Sergey Melnik, and Peter Mork. "Interactive schema translation with instance-level mappings". In: *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment. 2005, pp. 1283–1286.

[5] Yoan Chabot et al. "A Complete Formalized Knowledge Representation Model for Advanced Digital Forensics Timeline Analysis". In: *Digit. Investig.* 11 (Aug. 2014), S95–S105. ISSN: 1742-2876.

[6] Y. Chabot et al. "Automatic Timeline Construction and Analysis for Computer Forensics Purposes". In: *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*. Sept. 2014, pp. 276–279. DOI: 10.1109/JISIC.2014.54.

[7] *CISCO's 2014 Annual Security Report*. URL: https://www.cisco.com/web/offers/lp/2014-annual-security-report/preview.html (visited on 07/21/2015).

[8] Stefan Fenz, Thomas Pruckner, and Arman Manutscheri. "Ontological Mapping of Information Security Best-Practice Guidelines". In: *Business Information Systems*. Ed. by Witold Abramowicz. Vol. 21. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, 2009, pp. 49–60. ISBN: 978-3-642-01189-4.

[9] Almut Herzog, Nahid Shahmehri, and Claudiu Dumar. "An Ontology of Information Security". In: *International Journal of Information Security and Privacy (IJISP)* 1 (4 2007), pp. 1–23.

[10] John P. Mello Jr. "Cybersecurity Suffers from Talent Shortage". In: (Apr. 2015). URL: http://www.monster.com/technology/a/Cybersecurity-Suffers-from-Talent-Shortage (visited on 04/27/2015).

[11] Anya Kim, Jim Luo, and Myong Kang. "Security Ontology for Annotating Resources". In: *Proceedings of the 2005 OTM Confederated International Conference on On the Move to Meaningful Internet Systems: CoopIS, COA, and ODBASE - Volume Part II*. OTM'05. Agia Napa, Cyprus: Springer-Verlag, 2005, pp. 1483–1499. URL: http://www.nrl.navy.mil/itd/chacs/sites/www.nrl.navy.mil.itd.chacs/files/pdfs/Kim%20etal2005.pdf.

[12] Craig A. Knoblock et al. "Semi-automatically mapping structured sources into the semantic web". In: *The Semantic Web: Research and Applications*. Springer, 2012, pp. 375–390. URL: http://link.springer.com/chapter/10.1007/978-3-642-30284-8_32 (visited on 12/01/2014).

[13] Dongwon Lee et al. "Nesting-Based Relational-to-XML Schema Translation." In: *WebDB*. 2001, pp. 61–66.

[14] Lindsley G. Boiney. *The Human Side of Agile Cyber Defense: Leveraging Cyber Analysts' Expertise*. 2014.

[15] Alan W McMorran. "An introduction to iec 61970-301 & 61968-11: The common information model". In: *University of Strathclyde* 93 (2007), p. 124.

[16] Leo Obrst, Penny Chase, and Richard Markeloff. "Developing an Ontology of the Cyber Security Domain". In: *Proceedings of the Seventh International Conference on Semantic Technologies for Intelligence, Defense, and Security*. STIDS '12. Fairfax, Virginia, USA, 2012, pp. 49–56. URL: http://ceur-ws.org/Vol-966/STIDS2012_T06_ObrstEtAl_CyberOntology.pdf.

[17] Alessandro Oltramari et al. "Building an Ontology of Cyber Security". In: *Proceedings of the Ninth International Conference on Semantic Technologies for Intelligence, Defense, and Security*. STIDS '14. Fairfax, Virginia, USA, 2014, pp. 54–61. URL: http://stids.c4i.gmu.edu/papers/STIDSPapers/STIDS2014_T8_OltramariEtAl.pdf.

[18] *Ontology:DnS*. URL: http://ontologydesignpatterns.org/wiki/Ontology:DnS (visited on 06/08/2015).

[19] *Ontology:DOLCE+DnS Ultralite*. URL: http://www.ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite (visited on 06/08/2015).

[20] Simona Ramanauskaite et al. "Security Ontology for Adaptive Mapping of Security Standards". In: *International Journal of Computers Communications and Control* 8.6 (2013), pp. 878–890. URL: http://univagora.ro/jour/index.php/ijccc/article/view/764.

[21] Ansgar Scherp et al. "F–a Model of Events Based on the Foundational Ontology Dolce+DnS Ultralight". In: *Proceedings of the Fifth International Conference on Knowledge Capture*. K-CAP '09. Redondo Beach, California, USA: ACM, 2009, pp. 137–144. ISBN: 978-1-60558-658-8. DOI: 10.1145/1597735.1597760. URL: http://doi.acm.org/10.1145/1597735.1597760.

[22] *The CybOX Project*. GitHub. URL: https://github.com/CybOXProject (visited on 05/11/2015).

[23] *The MAEC Project*. GitHub. URL: https://github.com/MAECProject (visited on 05/11/2015).

[24] *The STIX Project*. GitHub. URL: https://github.com/STIXProject (visited on 05/11/2015).

[25] *The TAXII Project*. GitHub. URL: https://github.com/TAXIIProject (visited on 05/11/2015).

[26] *The Timeline Ontology*. URL: http://motools.sf.net/timeline/teimline.html (visited on 08/17/2015).

[27] Ju An Wang and Minzhe Guo. "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*. CSIIRW '09. Oak Ridge, Tennessee, USA: ACM, 2009, 34:1–34:4. ISBN: 978-1-60558-518-5.