

See discussions, stats, and author profiles for this publication at: <http://www.researchgate.net/publication/269633111>

Ontologies for Cyber Situational Awareness

CHAPTER · FEBRUARY 2015

DOWNLOADS

150

VIEWS

139

5 AUTHORS, INCLUDING:



Brian Ulicny

Thomson Reuters

39 PUBLICATIONS 76 CITATIONS

SEE PROFILE



Jakub Moskal

VIStology

12 PUBLICATIONS 13 CITATIONS

SEE PROFILE



Mieczyslaw M. Kokar

Northeastern University

163 PUBLICATIONS 1,639 CITATIONS

SEE PROFILE

Inference and Ontologies

Brian E. Ulicny^{1,*}

Email bulicny@vistology.com

Jakub J. Moskal¹

Email jmoskal@vistology.com

Mieczyslaw M. Kokar⁺²

Email m.kokar@neu.edu

Keith Abe⁺³

Email kabe@referentia.com

John Smith⁺⁴

Email jsmith@liveaction.com

¹ VISTology, Inc., Framingham, MA, USA
AQ1

Abstract

The importance of visualization—discussed in the previous chapter—does not diminish the critical role that algorithmic analysis plays in achieving CSA. Algorithms reason about the voluminous observations and data about the network and infer important features of the situation that help analysts and decision-makers form their situational awareness. In order to perform this inference, and to make its output useful to other algorithms and human users, an algorithm needs to have its inputs and outputs represented in a consistent vocabulary of well-specified terms and their relations, i.e., it needs an ontology with a clear semantics and a standard. This topic is the focus of the present chapter. We already touched on the importance of semantics in the Cognition and Technology chapter. Now we discuss in detail how, in cyber operations, inference based on ontology can be used to determine the threat actor, the target and purpose in order to determine potential courses of action and future impact. Since a comprehensive ontology for cyber security does not exist, we show how such an ontology can be developed by taking advantage of existing cyber security related standards and markup languages.

1. Introduction

The importance of visualization—discussed in the previous chapter—does not diminish the critical role that algorithmic analysis plays in achieving CSA. Algorithms reason about the voluminous observations and data about the network and infer important features of the situation that help analysts and decision-makers form their situational awareness. In order to perform this inference, and to make its output useful to other algorithms and human users, an algorithm needs to have its inputs and outputs represented in a consistent vocabulary of well-specified terms and their relations, i.e., it needs an ontology with a clear semantics and a standard. This topic is the focus of the present chapter. We already touched on the

importance of semantics in the Cognition and Technology chapter. Now we discuss in detail how, in cyber operations, inference based on ontology can be used to determine the threat actor, the target and purpose in order to determine potential courses of action and future impact. Since a comprehensive ontology for cyber security does not exist, we show how such an ontology can be developed by taking advantage of existing cyber security related standards and markup languages.

The common feature of cyber-security systems is that they need to react very quickly to a dynamic environment that changes its state independently of whether the human or computer agents act on it. The agents want to act on the environment so that its evolution, at least in the area of interest to the agents, leads to the satisfaction of their goals or, more likely, in the case of computer networks the avoidance of certain undesired states: e.g. infiltration, network compromise, and so on. Towards this end, the agents need to collect information about the environment (usually from many different sources), make decisions based on the collected information and their knowledge, act according to their decisions, collect feedback from the environment in response to the actions, and update their knowledge in order to make decisions in the future.

We use the term awareness as described in Kokar et al. (2009), i.e., in order to be aware

... one needs to have data pertinent to the objects of interest, some background knowledge that allows one to interpret the collected object data and finally a capability for drawing inferences.

The requirement for the capability of inference comes from such common-sense sources as the Webster's Dictionary: "*awareness implies vigilance in observing or alertness in drawing inferences from what one experiences.*"

In cyber systems, where the processing loops (Boyd, 1987) are very fast, much of inference must be performed by computers. In other words, automatic inference engines must perform the inference, which in turn requires that the information (facts) to be acted upon by such engines needs to be represented in a language with formal semantics. Inference engines take such facts as input and produce new information.

In the following, we first introduce a malware infection scenario and discuss how a human analyst would deal with the malware detection problem. Then we discuss how an approach that mimics the analyst's process is implemented using ontologies and an inference engine.

2. Scenario

In this chapter we consider a case of malware related to cyber espionage. The following describes significant events in the order of occurrence:

- On 1/1/2012 at 10 am an email message is sent to a particular user account, where the user is associated with a particular laptop (HP-laptop1), via yahoo with a PDF file attached containing malware that exploits a known vulnerability, CVE-2009-0658. This vulnerability causes a buffer overflow in Adobe Reader 9.0 and earlier. It allows remote attackers to execute arbitrary code via a crafted PDF document, related to a non-JavaScript function call and possibly an embedded JBIG2 image stream. This event is captured by Snort - an open source network intrusion prevention and detection system (IDS/IPS) developed by Sourcefire.
- On 1/1/2012 at 11 am a second email message is sent via yahoo with a PDF file attached containing malware that exploits the same vulnerability; the target is the same HP-laptop1. This event is also captured by Snort. Subsequently the user on HP-laptop1 opens the PDF file and the laptop gets infected.

- On 2/1/2012 the malware now installed on HP-laptop1 sends out a message via getPlunk.com to get the address of the Command and Control (C&C) server—a machine that can support the malware with the attack. GetPlunk.com is an intermediary micro-blogging service. This request is captured by Snort, which provides signature inspection, ID, and network event information.
- On 2/1/2012 the malware then uses the new C&C address to receive commands and exfiltrate data of interest to the attacker. This is captured by Snort based on the signature, ID and network event information.

3. Human Analysis of the Scenario

A human cyber analyst would need to examine network log files in order to detect a malware-supported botnet attack like the one described. In such a situation, a human analyst would need to:

- Detect the events mentioned among the network traffic logs. For example, somehow determine that one of many PDF email attachments that a particular laptop received contained malware.
- Determine that the laptop that opened the infected email attachment had a vulnerability that could be exploited on the basis of the installed software.
- Detect the exchange of messages between the infected laptop and the micro-blogging service that contains the address of the command-and-control computer address.
- Or, barring that, somehow detect an unusual amount or pattern of traffic between the infected laptop and the command-and-control server, despite the fact that every computer on the network interacts with multiple legitimate servers on a potentially regular basis.

The human analyst could make use of lists of known suspicious domains and IP addresses determined through looking up IP addresses in retracing the laptop's activity once it is discovered that the computer has become infected, by checking every incoming email with a PDF attachment over some period prior to the detection of the infection.

Without tool support, a human analyst would find it very tedious and time-consuming to trace back how a particular laptop came to be infected by malware that allowed the computer to be controlled by an external command-and-control server, if the analyst even became aware that the laptop was being controlled externally at all.

4. An Outline of the Use of Ontologies for Cyber Security

In this section we give a brief introduction to ontologies and automatic inference.

4.1. Ontologies

As used in the knowledge representation domain, the term “ontology” stands for an explicit, formal, machine-readable semantic model that defines the classes, instances of the classes, inter-class relations and data properties relevant to a problem domain (Gruber 2009).

In order for ontologies to be amenable to automatic processing by computers, they need to be represented

in a language that has both formal syntax and formal semantics. The syntax must be defined so that computers can recognize whether ontological statements (sentences) are grammatically correct. A formalized semantics means that a machine can judge whether two statements are consistent or not and imply one another or not. A vocabulary without a formalized semantics has at most a syntax: rules constraining what combination or strings of words in that vocabulary are part of that language. This is the case with most of the languages or protocols for exchange of information about the states of networks currently. They are purely syntactic.

For a vocabulary to have a formalized semantics, it must be possible for a computational machine to understand when a statement is true in that vocabulary (model theory), what can be inferred from a set of statements in that vocabulary (inference), and when it is impossible for a set of statements to be jointly true (consistency). An ontology is thus a logical description of a conceptual domain in a vocabulary expressed in a language with both a formal syntax and a formal semantics.

The W3C's (World Wide Web Consortium) Semantic Web activity provides today's most extensive deployment of interoperable vocabularies with semantics in the form of ontologies encoded as OWL (Web Ontology Language) Ontologies. OWL (W3C 2009) is the most commonly used language for expressing ontologies today and has by far the largest developer base. Therefore, we will focus exclusively on OWL ontologies in this discussion.

For our purposes, OWL represents individuals (e.g. elements of the network such as a particular router or a particular user), classes of individuals (e.g. "Router", "Printer", "Authorized User"), properties ("hasIPAddress", "hasPassword", "lastAccessed") that relate individuals either to other individuals ("object properties") or to datatype individuals (e.g. some date expressed as an xsd:dateTime).

The semantics of OWL is based on Description Logics (DL) (~~Jena n.d.~~ Baader et al. 2010), a family of Knowledge Representation (KR) formalisms equipped with a formal, model-based semantics. The architecture of a system based on DL allows for setting up a knowledgebase (KB) and for reasoning about its content. The KB consists of two components, known as the TBox and ABox. In the context of OWL, the TBox introduces axioms defining the ontology classes and properties and their relationship. For example, the TBox asserts relationships between classes. These can be simple class hierarchies (e.g. that every CiscoRouter is a Router) or arbitrary complex logical expressions (e.g. if a single send operation transmits a packet to multiple destinations, then it is a multicast operation). The TBox also enables the ontology to specify the domain and range of properties, cardinality restrictions on properties, property types (symmetric, transitive), property hierarchies, datatype range restrictions, and so on. We cannot provide a complete tutorial on OWL here, but its expressiveness is extremely powerful although not as powerful as ~~nd-comparable to~~ that of first-order logic.

The ABox contains assertions about specific named individuals—instances of classes and properties that have been defined in the TBox. In the context of cyber situational awareness, the TBox consists of axioms about the network domain, shared by all instances and states of the network. The ABox, on the other hand, represents facts pertaining to a particular network at particular times.

The logical definitions of the TBox combined with the assertions about particulars in the ABox allow an inference engine to identify new instances of a class by means of the properties it exhibits. Disjointness axioms about classes similarly allow the system to identify inconsistencies according to the ontology. For example, if only users of a certain class are allowed to access certain data, and a particular user is asserted as belonging to a class disjoint with the permitted class, yet there is an assertion that this user accessed that data, then an inconsistency will be detected in the ontology, and OWL inference will halt. A knowledge base must be consistent at all times, since anything can be inferred from an inconsistency.

OWL 2, the most recent version of the standard, defines three subsets, called *language profiles*, which are expressive and tractable, but tailored to specific needs:

- *OWL 2 EL*—terminological/schema reasoning, focused on terminological expressivity used for light-weight ontologies;
- *OWL 2 QL*—query answering via database engines, uses OWL concepts as light-weight queries, allows query answering using rewriting in SQL on top of relational DBs;
- *OWL 2 RL*—reasoning that can be implemented by standard rule engines.

OWL 2 RL, one of the most widely implemented profiles, is a collection of 75 implication and consistency rules. It has desirable characteristics: the set of entailed triples is finite and in PSPACE, and the runtime complexity is in PTIME.

In order to store and exchange OWL ontologies among applications, one of its concrete syntaxes is used. The primary exchange syntax for OWL is RDF/XML, which stems from the fact that the underlying model of OWL is RDF. Since OWL semantics can also be defined in terms of the semantics of RDF (Resource Description Framework) (n.d.), the serialization of OWL is a combination of the serialization of RDF and the additional concepts that are built out of RDF.

RDF's statements (sentences) consist of three elements—subject, predicate and object, collectively called triples. The predicate represents a relation, the subject is an element from the domain of the relation and the object is from the range of the relation. Thus conceptually OWL can also be viewed as a collection of triples.

Other concrete syntaxes, not based on XML, are often used as well, most notably Turtle, OWL XML, and Manchester Syntax (Wang et al. 2007). RDF/XML is the only required syntax to be supported by every OWL compliant tool.

While it may seem that OWL is just another XML language, because it can be expressed in XML, it is important to note that XML is just one of the syntaxes, while OWL has a formal semantics. Semantically equivalent documents in OWL can be expressed in multiple ways, using different syntaxes. Furthermore, because OWL is based on RDF, OWL knowledgebase can be queried with SPARQL, the query language for RDF.

Ontologies can be evaluated for their quality. Various evaluative dimensions are outlined in Brank et al. (2005), Obrst et al. (2007), Shen et al. (2006), Vrandečić (2009)). In Ye et al. (2007), the following criteria are proposed:

- **Clarity:** Concepts in an ontology should be uniquely identified and distinguished from other terms through necessary and sufficient conditions specified in the ontology.
- **Coherence:** Concepts in an ontology should be defined consistently.
- **Ontological commitment:** An ontology should be general enough to be usable by any application in that domain. It is advantageous that classes, associated properties, and involved constraints should serve for all of the general problems in the domain.
- **Orthogonality:** The defined concepts should be mutually exclusive from each other, which makes it

easier to share and reuse ontologies.

- **Encoding bias:** General ontologies should be independent of specific symbol-level encoding. That is, the names of concepts should not favor proprietary- or vendor-specific naming schemes.
- **Extensibility:** Ontologies should be extensible to allow them to be reused easily by other applications in a specific domain.

As cyber situation awareness ontologies emerge, they can be evaluated and compared along these dimensions.

4.2. Ontology Based Inference

Automatic inference on ontologies expressed in OWL is performed by *inference engines*, or *semantic reasoners*. An inference engine takes a set of facts about a specific domain of interest asserted in OWL and derives other facts using axioms and inference rules. In other words, an inference engine makes explicit the facts that are only implicit in the explicitly represented facts. The derived facts are logical consequences of the facts in a given fact base and ontology that is used to express the facts. Instantiations to any variables that were used in the derivation are also provided.

For instance, if the inference system knows that (in the context of a communication network) a device that routesPackets is a Router and that CiscoRouter routesPackets, then the system can infer that CiscoRouter is a sub-class of Router. While on the surface such an inference is simple and obvious, it is not obvious (although simple) for computers. Also, the derivation of such a fact may have far reaching consequences since now everything that was attributed to Router can be attributed to CiscoRouter as well. Other inferences can derive properties of sub-classes based on the properties of their super-classes (mentioned above), inferring that an individual is an instance of a specific class, and more. All kinds of inference types may be relevant to the case study discussed in this chapter, although the type of inference directly addressed will be the derivation of whether the agent's knowledge entails that a specific situation (malware intrusion) occurs.

OWL's semantics is such that inference is expected to be computationally tractable. OWL inference engines can therefore be guaranteed to make all inferences in a tractable amount of time. Thus, semantic scalability and interoperability can be achieved. The time for processing an OWL ontology depends on the size and the complexity of the axioms in the ontology. However, OWL's semantics guarantees that only valid inferences are drawn, i.e., inference engines are *sound*. At least some OWL profiles are such that *all* inferences can be made (such engines are *complete*) in polynomial time with respect to the ontology size. So the power of formal languages lies in the fact that they are equipped with a formal semantics, which guarantees that inference is sound, possibly complete, and in the case of OWL, tractable.

Several commercial inference engines for making inferences with OWL ontologies (ABox and TBox) are available (OWL/Implementations n.d.). Inference engines built to implement the OWL standard will infer all and only the same facts from the same set of given facts. In our project we used the BaseVISor inference engine (~~Barwise and Perry~~Matheus et al. 2006 +1983), which is free for research purposes. It allows for including additional inference rules.

4.3. Rules

The expressive power of OWL is relatively high—equivalent to a decidable fragment of complete First

Order Logic. The designers of OWL intentionally kept its expressiveness at that level in order to avoid the high computational complexity of inference. However, in many cases more expressive power than can be achieved with OWL is necessary. In such cases OWL can be supplemented with rules.

As an example, consider defining the class of routers that send the same packets. To achieve this, one would need to define this class as a set of pairs of routers, where one of them sends the same packet as the other. While OWL provides capabilities for linking, in this case, packets with routers, e.g., via a `sendsPacket` property, OWL will not allow to distinguish between the two routers and the two packets that are associated with the routers via this property. To achieve such a goal, variables are needed, a feature that OWL lacks. (For any two specific routers, OWL can be used to express that one router sends all and only the same packets as the second, however.)

The desired result can be easily achieved using rules, however. One can define a rule that would imply that whenever `sendsPacket(?R1, ?P1)` and `sendsSamePacket(?R1, ?R2)` then `sendsPacket(?R2, ?P1)`.

Issues with the expressive power of OWL versus rule languages have been long recognized and extensively discussed in the literature (cf. Horrocks and Sattler 2003). In particular, it has been recognized that OWL has serious limitations with handling complex role inclusions. For example, it is not possible to propagate the “ownership” role from an aggregate to its parts (sender of a frame is also the sender of all the packets in that frame). In our approach, we use rules whenever we encounter problems with the expressiveness of OWL alone. It should also be noted that just the use of rules per se does not guarantee the correctness of the solution; one needs to be aware of various pitfalls associated with the use of automatic inference and ensure that the implementation of the rules avoids such problems.

5. Case Study

The first requirement for this study was an ontology for the cyber security domain. Unfortunately, a comprehensive ontology for this domain does not exist. In this section we describe our approach to developing such an ontology. The ontology had to be supplemented with rules, which will be presented next. To demonstrate the usefulness of the ontology based approach to cyber security, a testing environment had to be set up, data collected and then used by an inference engine to infer whether a situation of a specific type of cyber threat is taking place or not.







5.1. Cyber Security Ontologies

An ontology for cyber situational awareness should be expressive enough to capture the classes and properties of individuals involved (e.g. networked devices, the persons or organizations who own, control, or use them), how these classes and properties are related to one another (including membership conditions, property domains and range restrictions, and disjointness axioms), and what elements are networked at particular times and the activities in which they are involved (e.g. message *M* was sent from element *X* to element *Y* at time *T*). On the basis of this representation of the network in a logical framework, a cyber situational awareness engine needs to characterize the situation as either operating normally or as being under a stage of attack or some otherwise undesirable condition.

Unfortunately, there have been relatively few attempts to develop an ontology to comprehensively encode cyber situation information as an OWL ontology (Swimmer 2008; Parmelee 2010; Obrst et al. 2012; Singhal and Wijesekera, 2010), and no comprehensive ontologies have been published. Some design patterns have been published for modeling important network concepts, such as the mereotopology patterns outlined in Dumontier (2013), however.

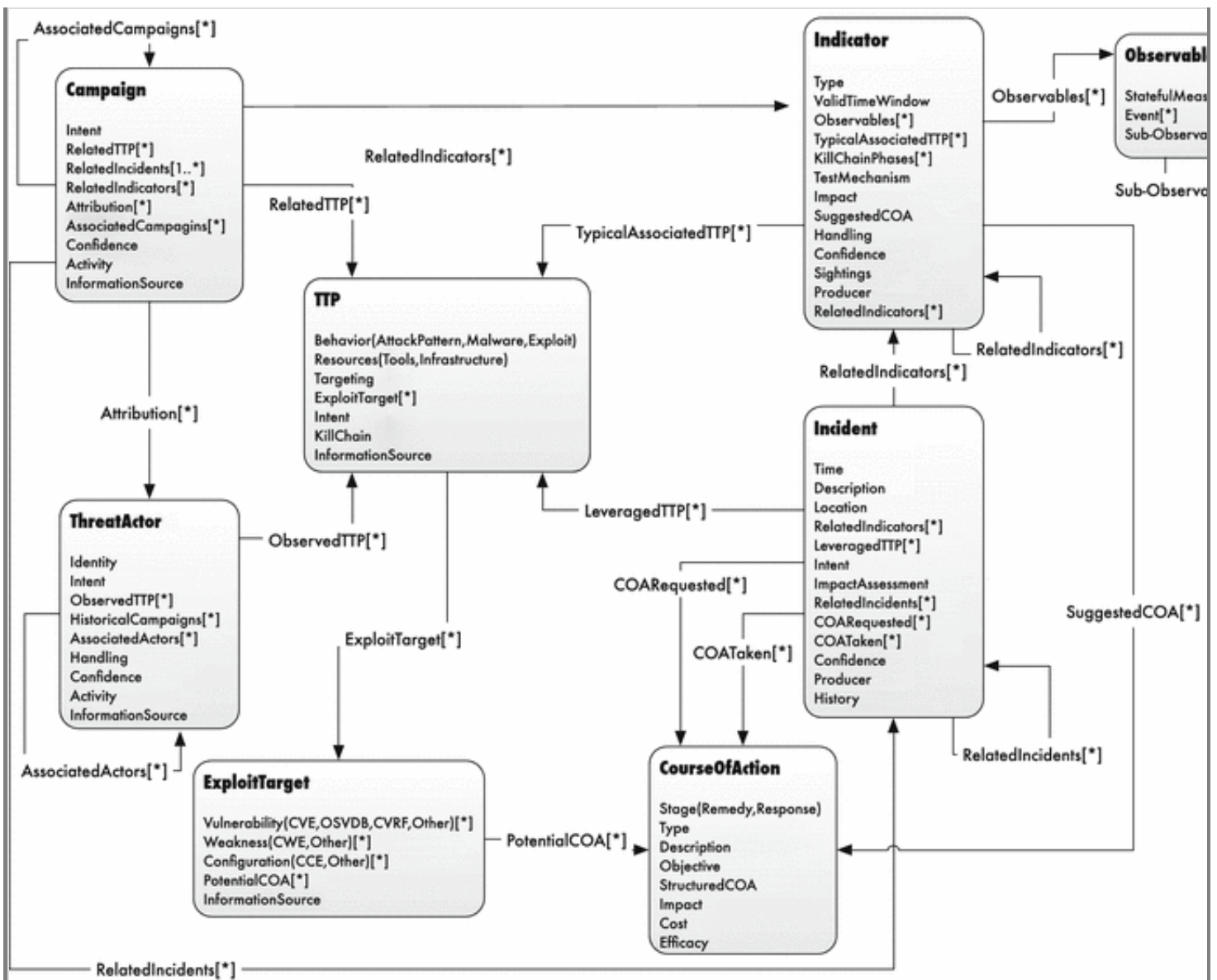
In our project, various cyber security standards from MITRE (CCE, n.d.), (CPE, n.d.), (OVAL, n.d.), (STIX, n.d.), NIST (SCAP, n.d.) and IETF (IODEF, n.d.) were investigated in representing the various aspects of cyber security. Figure 1 shows the major categories and the standards found in each category. To construct a working ontology for cyber situational awareness, these various standards were converted into OWL versions, in whole or in part. That is, the terms from the various XML-based vocabularies were integrated into an OWL ontology, thus providing them with a computer-processable semantics. Figure 1 lists the standards considered in the project and indicates whether or not they were converted to OWL as part of this effort.

Fig. 1
Security standards

	In OWL	Not converted
Incident and Event Reporting		
Threat Information		
Risk Information		
Assets, Target Information		

The most relevant standard is a relatively new effort supported by various organizations and managed by MITRE called Structured Threat Information eXpression (STIX) (n.d.). STIX is the most comprehensive effort to unify cyber security information sharing. It is described as “a Structured Language for Cyber Threat Intelligence Information”, and it incorporates vocabulary from several other standards. The overall structure is show in Fig. 2 . STIX is meant to allow for information sharing on cyber threat intelligence. An XML schema for STIX 0.3 was available and used to construct the initial ontology.

Fig. 2
Structured threat information eXpression (STIX) architecture v0.3 (Structured Threat Information eXpression n.d.)



STIX captures concepts such as those listed below and provides a high level framework to hold the various cyber intelligence components together. These include:

- Observable and Indicators
- Incident
- Tactics, techniques and procedures of attackers (TTP)
- Exploit Targets
- Courses Of Action
- Campaigns and Threat Actors

The STIX standard helps to glue together the lower level concepts such as events, device and the various other MITRE standards.

5.2. An Overview of XML-Based Standards

In order to reason over statements expressed in all of these vocabularies, we converted the following XML-based cyber standards into OWL ontologies, in whole or part.

5.2.1. Structured Threat Information eXpression (STIX)

STIX™ (n.d.) is a collaborative community-driven effort to define and develop a standardized language to represent structured cyber threat information. The STIX Language intends to convey the full range of potential cyber threat information and strives to be fully expressive, flexible, extensible, automatable, and as human-readable as possible.

STIX is sponsored by the office of Cybersecurity and Communications at the U.S. Department of Homeland Security.

5.2.2. Common Attack Pattern Enumeration and Classification (CAPEC)


CAPEC™ (n.d.) is international in scope and free for public use. It is a publicly available, community-developed list of common attack patterns along with a comprehensive schema and classification taxonomy. Attack patterns are descriptions of common methods for exploiting software systems. They derive from the concept of design patterns applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples.

CAPEC is co-sponsored by MITRE Corporation and the office of Cybersecurity and Communications at the U.S. Department of Homeland Security.

5.2.3. Common Vulnerabilities and Exposures (CVE)

CVE (n.d.) is a dictionary of publicly known information security vulnerabilities and exposures. CVE's common identifiers enable data exchange between security products and provide a baseline index point for evaluating coverage of tools and services.

5.2.4. Cyber Observables eXpression (CybOX)


 **The Cyber Observable eXpression** This paragraph should also be indented. (CybOX, n.d.) is a standardized schema for the specification, capture, characterization and communication of events or stateful properties that are observable in the operational domain. A wide variety of high-level cyber security use cases rely on such information including: event management/logging, malware characterization, intrusion detection, incident response/management, attack pattern characterization, etc. CybOX provides a common mechanism (structure and content) for addressing cyber observables across and among this full range of use cases improving consistency, efficiency, interoperability and overall situational awareness.

5.2.5. Malware Attribute Enumeration and Characterization (MAEC)


MAEC (n.d.) is a standardized language for encoding and communicating high-fidelity information about malware based upon attributes such as behaviors, artifacts, and attack patterns. By eliminating the

ambiguity and inaccuracy that currently exists in malware descriptions and by reducing reliance on signatures, MAEC aims to improve human-to-human, human-to-tool, tool-to-tool, and tool-to-human communication about malware; reduce potential duplication of malware analysis efforts by researchers; and allow for the faster development of countermeasures by enabling the ability to leverage responses to previously observed malware instances.

5.2.6. Common Weakness Enumeration (CWE)

 CWE (n.d.) provides a unified, measurable set of software weaknesses that is enabling more effective discussion, description, selection, and use of software security tools and services that can find these weaknesses in source code and operational systems as well as better understanding and management of software weaknesses related to architecture and design. Indent.

5.2.7. Whois and Additional Ontologies

 Several additional ontologies were used to represent organizations, whois information, and other data. Whois is a query and response protocol used for querying databases that store the registered users or assignees of Internet resources - domain names, IP address blocks, or autonomous systems. The protocol stores and delivers database content in a human-readable format. Indent.

5.3. Lifting Cyber Security XML into OWL

XML schemas merely mandate how information should be structured in conveying a pre-specified set of XML elements from one agent to another. What the various XML elements mean is implicit or, at most, only represented in the XML schema documentation, and implemented in code that inputs and outputs data in that schema, in a manner that is consistent or inconsistent with data encoded by other implementers. That is, following the distinctions outlined in the previous sections, the various XML schemas used to communicate cyber situations currently provide a set of concept names, with distinct URIs, and syntax (message format) but no formal semantics. Hence, they fall short of being ontology and cannot be subject to automatic inference.

The process of transforming XML data to semantic representation (RDF or OWL) is called *lifting*. This process can be largely automated, but with some caveats because not all XML schema constructs have a direct counterpart in OWL. A plethora of research has been conducted in this area (Bedini et al. 2011a; Bikakis et al. n.d.; Anagnostopoulos et al. 2013; Ferdinand et al. 2004; Bohring and Auer 2005; Rodrigues et al. 2006).

In converting the XML-based schemas to OWL, we faced the following choices:

- Use a generic XSLT Translation: Since both XSD and OWL are XML-based formats, XSLT can be used to transform XSD to OWL.
- Write a custom XSLT script: This approach requires writing a tailored XSLT script per XSD allowing the resulting OWL to more accurately represent the schema.
- Write custom application: This is a more heavy-duty solution that involves writing a procedural code (Java, C++, etc.) that loads the specific XSD document into memory and generates OWL (possibly using an OWL library).

- Use an external tool: Use one of the tools available on the market, like TopBraid Composer.
- Manual: Manually create the ontology based on the XSD data model using an ontology editor such as Protégé.

The STIX XML Schema is quite complex. It uses many advanced features, like `xsd:choice`, `xs:restriction`, `xs:extension`, `xs:enumeration`, which often cannot be easily transformed into OWL. Moreover, the model is partitioned into multiple files that not only import each other, but also import additional external schemas. Not surprisingly, the automatic translation offered by generic XSLT scripts or by the third party applications did not yield satisfactory results. The resulting ontology was inconsistent, messy and effectively very hard to use. Enumerations were improperly converted, namespaces were improperly defined, and countless classes and properties that were generated were practically useless, acting as placeholders. Numerous classes and properties were redefined across multiple generated OWL files, which produced inconsistencies. Moreover, the translation did not handle datatypes correctly and generated object properties when it was not necessary, or correct, effectively defining primitive data types as OWL classes, such as String.

In general, since XML Schema is concerned only with message structure, not message meaning, datatype properties such as strings are used to encode entities, making OWL inference impossible over them, since they do not represent entities (i.e. something denoted by a URI/IRI) but only a piece of data. For example, if the country associated with an IP address is represented as a string in OWL (e.g. “Republic of Ireland”), then it is impossible to do geospatial reasoning with this element using an ontology of places and their relations ([GeoNames Ontology, n.d.](#)). In order to reason about something like a region, the entity must be a first-class individual with associated properties (latitude-longitude coordinates, etc.), denoted by an IRI/URI.

Automated translation methods were not yet satisfactory for use in converting the STIX model to OWL. Techniques for such automatic translation in general are still subject of an ongoing research (Bedini et al. 2011 b). In general, automated XSD to OWL methods fail because they cannot:

- Distinguish container elements (e.g. Contacts) from singular classes to which a parent class might be related one-to-many (e.g. Contact).
- Distinguish datatype and object-type properties.
- Generate useful object property relation identifiers.
- Produce useful property restrictions between classes (except for cardinality restrictions, which were often accurate).
- Generate domain and range restrictions.
- Generate useful class hierarchies.

Our second approach was to write custom XSLT scripts. However, due to the complexities of the STIX model described above, the result was also far from satisfactory, and we decided to create the STIX ontology by hand, using the XSD model as a guideline. During the manual process we were able to leverage some OWL-specific mechanisms and create an ontology that maintained the intention of the schema, but used slightly different constructs to express it. For instance, in order to associate an indicator with a sequence of observables in XSD, authors of the STIX model created a type `stix:ObservablesType`, which is a complex type with an unbounded sequence of `stix:ObservableType`. To represent the same in

OWL, only one class is needed, e.g. `stix:Observable`, whose instances can then be related to instances of a class `stix:Indicator` using an object property “`stix:observable`”. If we wanted to restrict this relationship, e.g., say that a single `Indicator` can have only one `Observable`, we would simply define the “`stix:observable`” property as a functional property.

Nearly all relationships between XSD elements were expressed as OWL restrictions on classes that represented one of the elements. For instance, in order to express the fact that an instance of `stix:Indicator` can have a relationship with multiple instances of `stix:Observable`, we defined a restriction on class `stix:Indicator`:

```
<owl:Class rdf:about="http://stix.mitre.org/STIX#Indicator">
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="http://stix.mitre.org/STIX#observable"/>
<owl:allValuesFrom
rdf:resource="http://stix.mitre.org/STIX#Observable"/>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

In the end, the resulting ontology reflects the relationships expressed in the original model, yet it is clean, logically consistent and easy to use when writing SPARQL queries or declarative rules.

5.4. STIX Ontology

The resulting STIX OWL ontology that we created on the basis of the STIX XML Schema (XSD) document has the following characteristics (Fig. 3):

Fig. 3

STIX 0.3 ontology metrics

METRICS		CLASS AXIOMS	
Axiom	527	SubClassOf axioms count	270
Logical axiom count	304	EquivalentClasses axioms count	11
Class count	67	DisjointClasses axioms count	0
Object property count	83	GCI count	0
Data property count	33	Hidden GCI count	6
Individual count	11		
DL expressivity	ALUHOQ(D)		

AQ2

The STIX ontology imports the following ontologies, in addition to the multiple STIX ontology files (Fig. 4):

Fig. 4

STIX ontology imports

PREFIX	NAMESPACE
Common	<i>http://cybox.mitre.org/Common#</i>
capec_v1	<i>http://capec.mitre.org/capec_v1#</i>
data-marking	<i>http://data-marking.mitre.org#</i>
cybox_v1	<i>http://cybox.mitre.org/cybox_v1#</i>
killchain	<i>http://referentia.com/killchain/</i>
maec-core-2	<i>http://maec.mitre.org/XMLSchema/maec-core-2#</i>

AQ3

A high-level overview of the STIX ontology that was produced in this project, is depicted in Fig. 5. The central class is TTP (Tactic, Techniques and Procedure), the instances of which are cyber security exploits. These TTPs may have a variety of subclasses. TTPs are related to instances of AttackPattern, Exploit, ExploitTarget, InformationSource, Infrastructure, Intent, KillChain, KillChainPhase, Identity (of target) and ToolInformation via object properties that further individuate the instances of that class. TTP has subclasses ObservedTTP, RelatedTTP, and LeveragedTTP. Indicators are related to TTP by means of the indicatedTTP object property. Restrictions on the object properties are used to structure the instances of this class, and other classes.

Fig. 5

High-level view of STIX ontology



5.4.1. Populating Vulnerabilities Using the NVDCClient

The NVDCClient is a tool we have developed that uses the National Vulnerability Database (available from NIST n.d.) to fetch data about a particular cyber vulnerability in the CVE (Common Vulnerabilities and Exposures) registry (CVE n.d.). The data includes information about the CVE's impact (Common Vulnerability Scoring System CVSS n.d.), references to advisories, solutions and tools and a relationship to CWEs (n.d.). Because they were so numerous, we did not want to first incorporate all the vulnerabilities into an ontology that we would use in inference, since there would be too much information that was not necessary. The information in the CAPEC ontology relates attack patterns to classes of vulnerabilities.

NIST does not provide any web service access to its data; instead, it publishes a data feed, which is regularly updated. The feed is formatted in XML and is rather large (16 MB). Downloading and processing the entire feed is impractical if one is interested in getting information about only a few CVEs at a time. To address this issue, we developed a Java program that scrapes the CVE information directly from its corresponding web page. All CVEs are described on single pages accessible under an HTTP address of the following format:

`http://web.nvd.nist.gov/view/vuln/detail?vulnId= <CVE ID>`

Consequently, the tool downloads the HTML code of the web page describing a particular CVE of interest by supplying appropriate *CVE ID*. Next, using the jsoup library (jsoup n.d.), it scrapes useful information about the CVE using a CSS-like syntax to access particular HTML elements on the page.

Once the CVE information is stored in memory, the tool creates a new ontology model in Jena (n.d.) and

generates an OWL document representing the CVE. Tools like TopBraid Composer or BaseVISor inference engine can then process the resulting OWL document, which are imported at inference time.

5.5. Other Ontologies

5.5.1. Persons, Groups and Organizations

We used the Friend of a Friend (FOAF) ontology (The Friend of a Friend n.d.) to represent persons, groups, and organizations. Additionally, we developed an ontology for WhoIs Internet registration information (WhoIs n.d.) in order to associate IP addresses of devices with human agents and organizations. We used a service called whoisxmlapi.com to provide WhoIs information for IP addresses we encountered. This requires two lookups. The first lookup identified a contact email based on the IP address supplied. We then did a DNS lookup on the contact email domain (assuming they would be the same) to get the domain name and other information for the domain that controls that IP address. Similarly, we could look up the registrar name. If the registrar is suspicious, knowing this fact about an IP address is useful for our purposes. Unfortunately, the XML produced by the whoisxmlapi.com service was less useful for foreign IP addresses, because it failed to contain structured information. Relevant information was contained in the record, but it was enclosed just as text strings, often in a foreign language, which would require additional parsing to populate the XML fields and then lifted to OWL. We did not implement a custom parser for accomplishing this.

5.5.2. Threat Representation

For threat representation, the NIST CAPEC classification was used and represented in an ontology. The CAPEC classification provides ways to classify the attacks and used with detected cyber events to understand the CVE vulnerabilities, which were associated with an exploit using a particular CAPEC attack mechanism. This was then used to infer potential effects on the target and future actions the malware may perform.

The MAEC schema was not used as it is typically used to represent the actual malware and this type of data is usually not accessible at various security sites, unless through a paid subscription. CVE and vulnerability information is typically available but not the actual malware data.

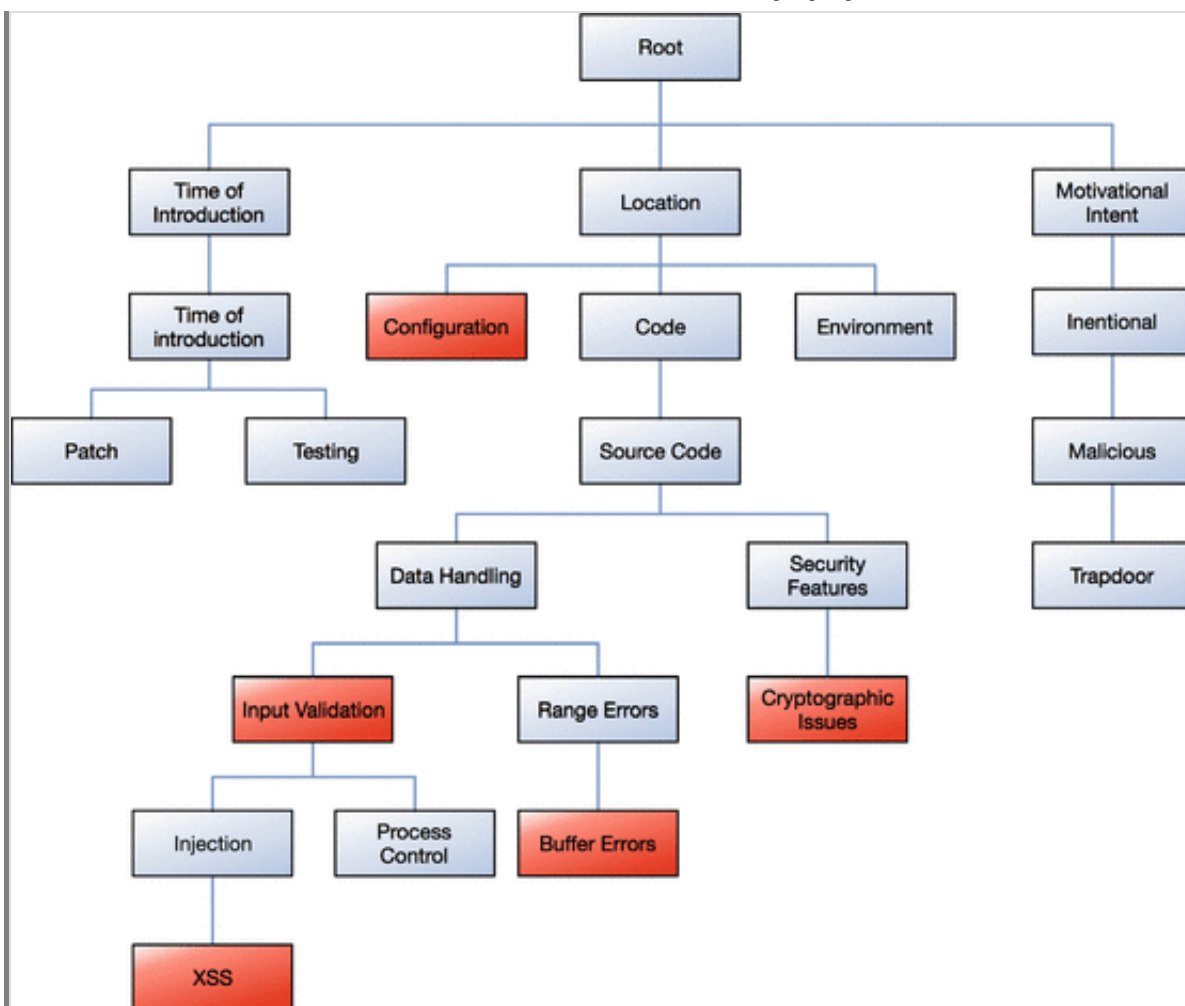
5.5.3. Weakness and Risk Representation

The CWE representation that classifies the various types of weaknesses was entirely represented in the ontology with the referenced CAPEC information. CVE vulnerability information was not represented in the ontology, as it is usually part of the signature used in detecting security events. Typically a Snort signature would identify a specific CVE, which would then be used to identify associated CWE and the CAPEC information.

The CWE provides the various weaknesses that may be exploited by malware and can be used to understand the attack and also determine the impact and malware behavior. Figure 6 depicts a portion of the CWE structure; the red boxes represent the CWEs that are being used by the National Vulnerability Database.

Fig. 6

Portion of the structure of the common weakness enumeration (CWE) (CWE n.d.)



5.5.4. Asset and Target Representation

MITRE standards such as CPE, CCE and CVSS were used to represent the target and asset being targeted by malware. The CPE and CCE tend to be more for end host representation of the configuration, operating system and applications but are not used in the ontology, as those concepts already exist. The CVSS information on the vulnerability score will be used for specific attacks to understand the potential impact and severity of the attacks.

Other ontologies that were developed include an ontology of Snort events and similar events, an ontology of part-whole relations, that were used to represent IP domains and sub-domains, and a watchlist ontology. The prototype system was capable of processing events from various sources and converting them into the ontology. For Snort IDS events, the system queried the Snort database and processed the various events. The key Snort field was the classification of the type of event and for known signatures, the CVE code of the exploited vulnerability.

For associating users with entities on the network, an LDAP directory was queried as to domain login and logoff information, which were converted into ontology events.

Netflow events were obtained from an existing netflow collector; since the system was not producing a high level of netflow data, occasional processing of the flow to ontology was done. Performance data of the network was obtained from a network management capability that polled SNMP related statistics and converted it into OWL form.

5.5.5. Kill Chain

Next, we discuss issues related to the modeling of events and situations in an ontology, representing and reasoning about the participation of entities in events and situations as well as mereological, causal, temporal and geospatial relationships between events. An ontology of events or situations treats these entities as individuals that occur or happen. They are considered to be perduring entities that unfold over time, i.e., they have temporal parts. In contrast, material objects such as stones and chairs are said to exist, and all of the parts that they have exist at each point in time; they are called endurants. There are several ontologies for events (Wang et al. 2007; Raimond and Abdallah 2007; IPTC 2008; Doerr et al. 2007; Mueller 2008; Francois et al. 2005; Westermann and Jain 2007; Scherp et al. 2009) and models for situation-awareness that have a close relation to events (Chen and Joshi 2004; Wang et al. 2004; Yau and Liu 2006; Lin 2008), some of which are based on Barwise and Perry's Situation Theory (Barwise and Perry 1983; Matheus et al. 2005, 2003; Kokar et al. 2009) that can be used to represent occurrences within a network.

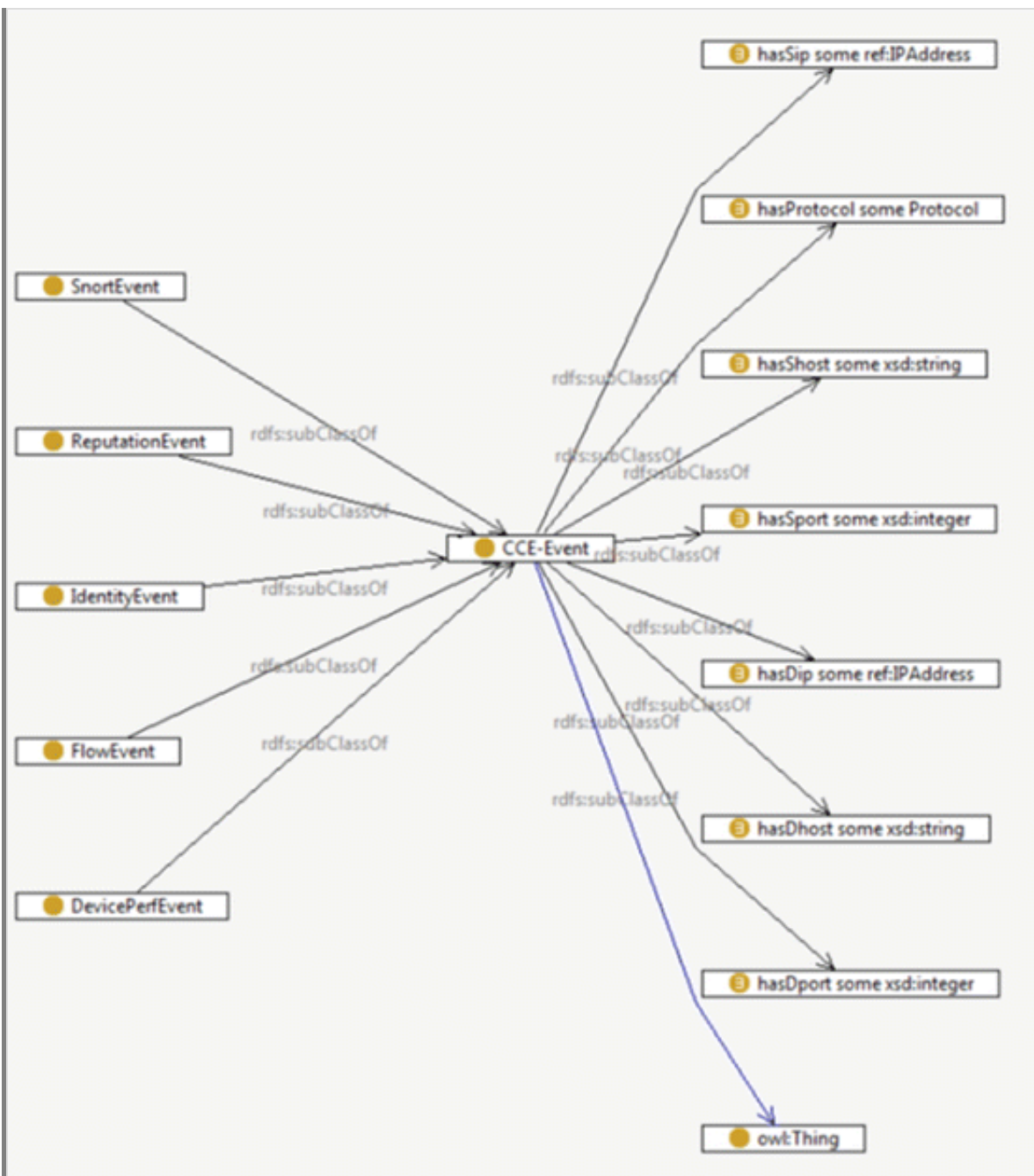
In our system, an Event class based on the MITRE Common Event Expression (CEE) was used to capture the low level events data that is captured in typical log data. The base ontology uses some of those concepts including time (Hobbs and Pan, 2004), priority and other common attributes found in CEE. The ontology was further enhanced to capture the following event subclasses:

- Security Events
- Identity Events
- Reputation Events
- Network Flow
- Network Performance Events

Figure 7 shows the ontology class structure from the ontology tool.

Fig. 7

Event ontology structure



One of the key cyber situational concepts from STIX is the idea of the kill chain, which is a particular type of situation, in which several events of particular types must occur in a specified order. Kill chains as they are related to Advanced Persistent Threats are described in detail in Hutchins et al. (2011). The kill chain model of Advanced Persistent Threats is used to describe the various phases that an adversary would perform as listed below and shown in Fig. 8:

- Reconnaissance
- Weaponization
- Delivery
- Exploitation

- Installation
- C2
- Actions on Objectives

The kill chain referenced in STIX is slightly different than described in the Hutchins paper but similar concepts are expressed. An Intrusion Kill Chain, in our ontology, is a subclass of Kill Chain that contains various associated phases for that type of kill chain (here, Intrusion), which are related to one another by a transitive ‘precedes’ object property.

Fig. 8

Kill chain model (Security Intelligence n.d.)



AQ4

6. APT Test Use Case

For testing purposes, an APT type example was selected using a recent incident that was described in Stewart (2013). An “Advanced Persistent Threat” (APT) refers to cyber-espionage activity carried out against governments, activists, and industry. The entities involved could be anything that provides a point of information—malware, command and control (C2) domains, hostnames, IP addresses, actors, exploits, targets, tools, tactics, and so on. The “advanced” aspect is that sophisticated techniques using malware are used to exploit vulnerabilities in systems. For example, APTs may use sophisticated “ransomware” like the CryptoLocker exploit that attack a computer by means of an email attachment which then encrypts the data on the machine and demands that the user pay a ransom to unencrypt it (US-CERT 2013). The “persistent” aspect is that an external command and control system is continuously monitoring and extracting data from a specific target. The “threat” aspect refers to adversarial human involvement in orchestrating the attack.

An APT event contains various events that happen over a period of time that encompass various portions of the kill chain. The number of entities to keep track of is huge: there are hundreds of unique families of custom malware involved in APT cyber-espionage campaigns, using thousands of domain names and tens of thousands of subdomains for hosting malware or spearphishing.

At the most basic event level, reasoning about an APT using a cybersituational ontology requires detecting low-level network events, consisting primarily of Snort and Netflow events. In our scenario, these events were used to infer high-level concepts using the inference and rules in the ontology to generate individuals such as observables related to the events, which then become indicators and are represented as incidents in the ontology.

Mid-level entities were then asserted for correlation to higher-level concepts such as, campaign, TTP, threat actor and targets by means of inference using the ontology and custom inference rules that cannot be expressed in the ontology directly.

The rules that were developed were used to infer the various STIX classes and how they were related.

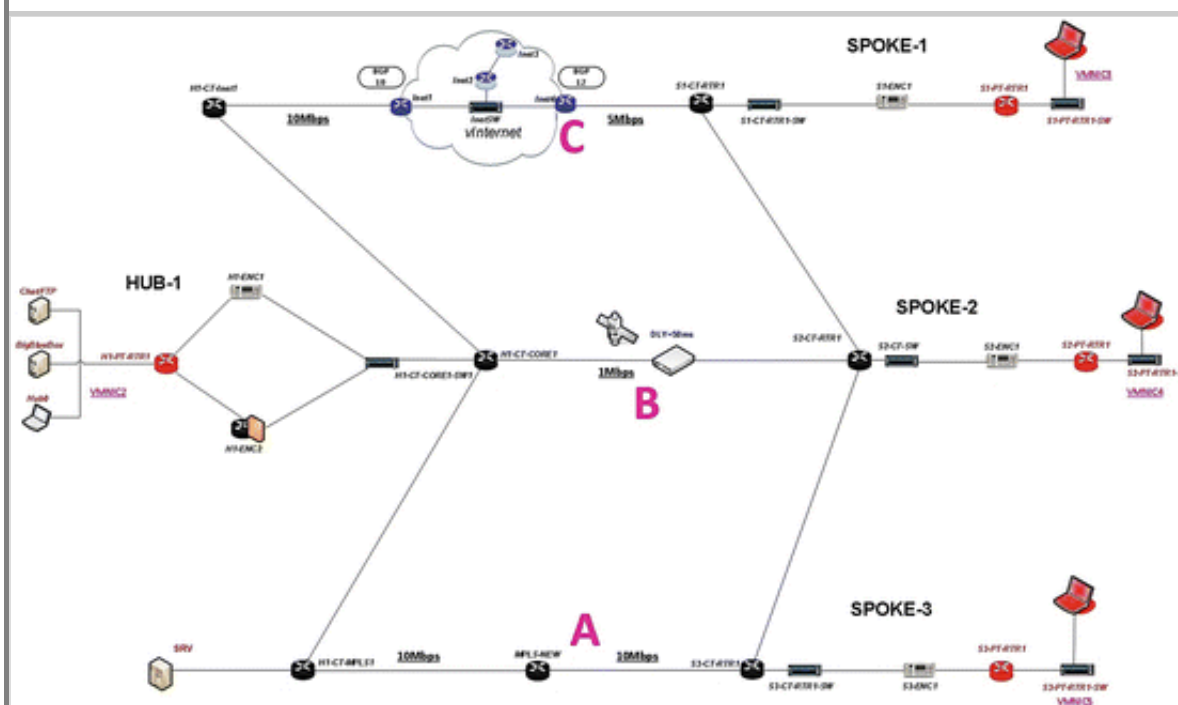
The rules would be used to infer the kill chain, kill chain phases, TTP, threat actor and target. Inferences were made associating various detected events with kill chain stages.

6.1. Test Network

Testing was done on a test network with actual attacks. The network consisted of a hub site connecting to three spoke sites (Fig. 9). There are three potential paths to the spoke sites with various IDS, firewall, router and switches. Various attacks were performed on the network and event information captured and analyzed by the prototype system. The ontology and rules were then used to infer information about the attack and its impacts on the test network. The inference of facts triggered by low-level Snort events to the identification of kill chain elements to the identification of perpetrators, was done using VISTology's BaseVISor OWL 2 RL forward-chaining inference engine, incorporating the cyber situational ontology produced from the various standards described here.

Fig. 9

Test network diagram. Some details are redacted



The inference and rule development consisted primarily of (1) creating the high level STIX incidents from various low level network events, and (2) inferring information as to the state of the network and its paths and potential impacts.

The STIX representation requires development of rules that build out a model of the cyber security state from the network devices, topology, assets with dynamic event information from IDS, NetFlow, network metrics and other event data.

The STIX representation creates STIX incidents, campaign and other high level classes based on individual events. Individual interrelated events such as the CVE Snort based event and the Snort class type were then pieced together to understand the attack type. The IP address from the Snort event was used to infer the identity of the APT threat actor.

6.1.1. Kill Chain

The various low level incidents were correlated as part of a larger kill chain. Looking at the user and target, and correlating the order of various incidents, was used to determine that these events were part of a similar campaign by the same threat actor. The temporal order of the events was then used to classify the nature of the situation. For example, if C2 beaconing occurs before an infected file is downloaded, then this is reversed from the kill chain sequence, so then these two incidents are not stages of the same kill chain, although they may be separate elements of attempts to inject an infected file.

6.1.2. Target Information

The IP address from the Snort sensor was used to understand the targeted devices and user identity from the identity events by means of WhoIs information. Observations about domain registrants were used to associate various cyber events with the same user. As target, information represented in the ontology about a user's role and place in an enterprise can be used to infer actual and potential attack types and vulnerabilities exploited.

6.1.3. Threat Actor

In our ontology, a threat actor reputation list was built based on the reputation list but also from external linkage to WhoIs and domain information sites. Eventually other external incident reports will help to correlate threat actors to larger campaigns that target multiple organizations. It may be possible to infer the threat actor type (e.g. state sponsored, criminal or individual hacker) based on TTP, target and sophistication of attack methods.

Reputation events were researched in conjunction with the Collective Intelligence Framework (CIF) project as a potential source of external IP- and URL-based reputation information. CIF uses data from sites such as the Malware Domain Blocklist, Shadowserver, Spamhaus, and several others, to store and correlate data. For our purposes, we just used Spamhaus (spamhaus.org) reputation lists directly. In the future, the interface to the CIF could be used to get additional information.

6.1.4. Impact Analysis

Rules were developed to infer information about the state of the network and the paths within the network based on the current attacks with respect to impact and mitigation.

In general, our ontology was augmented with some rules used to help infer the more complex APT concepts and to create instances of the STIX representation of the potential attacks. The rules were written in VISTology BaseVISor language and run with the BaseVISor inference engine. The inference engine runs all the rules to infer from the low level events to STIX observables, indicators and incidents in addition to standard OWL 2 RL inference rules for generic inferences about classes and relations. From these low level indicators, a kill chain with a kill chain phase was inferred based on the potential threat actor and target.

SPARQL queries were also used for analysis of impact based on various Snort, reputation and identity events. The Snort events that had a CVE reference were correlated to the CWE and CAPEC associations in the ontology to infer additional network impact scores.

6.2. Rules

In addition to the ontology, a set of rules was developed in order to make the following inferences:

- Tie Snort events to CVEs
- Create an Observable for each Snort or Netflow Event
- Create an Indicator for each Observable
- Extract Kill Chain phase from Snort events to Indicators
- Combine Indicators with the same related CVE
- Combine Indicators with the same signature IDs and source IP
- Combine Indicators with Snort events and NetFlow events of the same source/destination IP and detect time
- Combine indicators with NetFlowEvents matching source/destination but not having a corresponding SnortEvent
- Assert kill chain phase for NetFlow-based Indicators missing a kill chain phase
- Create Incidents for Indicators
- Extract CVSS Base score
- Identify ExploitTarget (destination of Snort event in the user's domain)
- Create a TTP and ThreatActor based on existing ExploitTarget
- Extract AttackPattern from ExploitTarget's CWE to ThreatActor's CAPEC
- Store IP addresses of ThreatActor based on events from ExploitTarget
- Assign indicators as KillChainPhase

These rules were implemented in BaseVISor rule language as Horn clause rules that enable patterns of triples to be expressed using both constants and variables in the body of the ~~text~~rules. The values of variables that are bound to a set of facts that matches the pattern in the body of the rule can then be used to assert new triples in the head of the rule. Additionally, these values can be used as inputs to procedural attachments in the rule head. Such procedural attachments might, for example, call external services to convert a domain name to an IP address or compute the distance between two geospatial points expressed as latitude/longitude pairs.

When security events are found by Snort, if the signature has a CVE associated with it, that information is used as a key to get additional information about the particular vulnerability from the NIST site. The NIST site provides html based data on the CVE but also the association to the CWE, CAPEC and additional information such as CVSS base score, access vector, and so on.

At a basic level, the detected Snort and Netflow events contain CVE signature, host and destination IP address and host information. The events are collected together as observables, which are mapped to indicators by means of the ontology and rules and are then aggregated as Incidents. Snort events with CVE were used to infer which part of the kill chain an event may be characterized as by mapping Snort event classes to kill chain phases in the ontology. Snort events without an associated CVE are correlated with Netflow events to represent their relation. For example command and control traffic may show up as a suspicious event in Snort but with no associated CVE. Such events can then be correlated with Netflow events on the basis of source and destination information, in order to infer matches to a particular target and threat actors. These mid-level Incidents are then associated with higher-level concepts of a campaign, TTP, threat actor and targets by means of the ontology and rules.

6.3. Inference Based Threat Detection

Rules were used to associate domains or IP addresses with incidents. Multiple events with the same source could then be detected. If the event was characterized as malevolent or part of a kill chain, the domain or IP address could be associated with entries in the reputation ontology derived from the Spamhaus.org entries described previously. In addition, ontology individuals corresponding to countries, organization and Internet registrars allowed the system to characterize domains and IP addresses as more or less suspicious, based on prior activities. Countries can be characterized by their reputation for hosting cyber aggressors, for example. IP addresses can be related to countries by means of WhoIs and DNS lookups, as described previously, as well. Thus, suspicious traffic originating from a known suspicious country would be inferred to be even more suspicious in nature.

7. Other Ontology-Related Efforts for Cyber Security

A number of research papers were found reporting on the use of ontologies in cyber security. First of all there are a number of papers that argue for the need to use ontologies in solving cyber security problems. For instance, in an ARO workshop presentation, Sheth (2007) argued for the use of the Semantic Web techniques and ontologies for cyber situational awareness. Caton (2012) argued for putting the issue of cyber security in a wider context of a more general theory of conflict—going beyond the models of the current situation—to models that include the capabilities of accommodating future developments in the cyberspace warfare. Clearly, this view is in line with the Endsley's model of situation awareness (Endsley 1995) in which the process includes projection in the future. Atkinson et al. (2012) put the problem of cyber security in a wider perspective of social networks and argued for an approach in which the problem is considered as a problem of establishing a cyber-ecology in which means exist for encouraging good behaviors and deterring bad. Their ontological framework includes both the social trust and technological rules and controls.

Ontologies for cyber security go back to the early days of the Semantic Web. For instance, the 2003 paper (Undercoffer et al. 2003) discussed the use of the DAML language (the precursor of OWL) for representing an ontology for the domain of intrusion detection. It compared DAML vs. XML and discussed the inadequacies of the latter. The ontology includes 23 classes and 190 properties/attributes. The 2005 paper (Kim et al. 2005) presented some plans to use ontologies for modeling dependencies between cyber related infrastructures, although no details of the resulting ontology were given. More et al. (2012) describe an experimental system for intrusion detection that uses ontologies and inference. Their ontology is an extension of the one developed by Undercoffer et al. (2003). Okolica et al. (2009) develop a framework for understanding situational awareness in the cyber security domain. They also refer to the ontology developed by Undercoffer et al. (2003).

A large number of papers reported on the use of ontologies for cyber security and situational awareness. However, since the details of the ontologies used are not shown, it was impossible for us to reuse these results. For instance, the paper (Khairkar et al. 2013) mentions a number of research efforts where ontologies were used. It claims that ontologies will resolve many problems with the Intrusion Detection Systems, in particular by classifying malicious activities detected in the logs of network activity, however, it does not show any details of such an approach. The paper (Kang and Liang 2013) discusses an attempt to use an ontology that links security issues to the software development process based on Model Driven Architecture (MDA). In particular, an ontology (which is presented as a collection of the various ~~meta~~meta-models) is attributed with the generation of the various MDA models. Unfortunately, the paper does not show any details of how this is achieved. Strassner et al. (2010) discuss an architecture of a system for network monitoring in which ontologies are used. The paper stresses the ability of ontologies to infer facts that are not explicitly represented. But no details of the ontology are shown. Bradshaw and his colleagues (2012) discuss the use of policies and an agent based architecture for the tasks of cyber situational awareness focusing on the human-computer interoperation. Oltramari et al. (2013) discuss the use of ontologies for decision support in cyber operations, however no specific ontology is provided. The paper focuses on the architecture and conceptual justification for the use of cognitive architectures that should include ontologies. The authors of Barreto et al. (2013) describe a system that uses ontologies. They don't show a new ontology, but rather reuse other existing ontologies.

A number of papers show a (usually graphical) representation of the used ontology. The paper (D'Amico et al. 2010) reports on a workshop in which participants worked on an ontology for capturing relationships between missions and cyber resources. This work is part of the Camus project (cyber assets, missions and users) (Goodall et al. 2009). In Strasburg et al. (2013) a project is described in which an ontology, expressed in OWL, was used to represent the domain of intrusion detection and response. The paper shows only the top level of the ontology. The paper (Bouet and Israel 2011) discusses a system in which an ontology for describing assets and their security information is used. The system works off-line —auditing long files. The paper shows only the top level of the ontology.

By contrast, Fenza et al. (2010) used the Situation Theory Ontology (STO) (Kokar et al. 2009) (which is publicly available) for identifying security issues in the domain of airport security.

Some papers focus on the process of developing ontologies for the cyber security domain. These were of special interest to us since we had to undertake such a task, too. For instance, Wali et al. (2013) describes an approach to developing a cyber security ontology from a cyber security textbook index and an existing ontology (Herzog et al. 2007). The Software Engineering Institute's report (Mundie and McIntire 2013) describes the Malware Analysis Lexicon (MAL)—an initiative influenced by the JASON report (McMorrow 2010).

We close our literature survey with a relatively recent report from MITRE (McMorrow 2010). JASON, an independent scientific advisory group that provided consulting services to the U.S. government on matters of defense science and technology, published a very influential report (McMorrow 2010) in 2010 whose objective was to examine and assess the theory and practice of cyber-security. One of the most important conclusions of this report was:

The most important attributes would be the construction of a common language and a set of basic concepts about which the security community can develop a shared understanding. Since cyber-security is science in the presence of adversaries, these objects will change over time, but a common language and agreed-upon experimental protocols will facilitate the testing of hypotheses and validation of concepts.

8. Lessons and Future Work

Given the complexity of the current XML-based standards and the state of the art in automatically converting XSD to OWL, the semantic representation of cyber threat information in an interoperable format that can be reasoned over (such as OWL) is difficult. Keeping up with changing and additional standards requires a great deal of manual knowledge representation effort.

In the test scenario described here, the ability of ontologies to represent information and infer additional information was used to identify an advanced persistent threat (APT) whose operation consisted of multiple steps, represented in our ontology as sequential steps in a kill chain. Inference with the ontology was used to understand the threat actor, the target and purpose, which helped to determine potential course of action and future impact.

In general, our experience with using existing terms and concepts implicit in XML-based cyber standards to create an OWL ontology with rules for inferring cyber situational awareness led us to the following conclusions:

- Ontology based analysis is useful for providing cyber situational awareness due to its ability to find patterns and infer new information by integrating information from a number of sources expressed in different standards.
- The ability to tie different event types to infer incident information is promising and could be used to add additional event types in the future.
- Automatically generated ontologies from XML schemas require a lot of massaging to be useful and create a lot of complexity that is difficult to fix.
- The MITRE security standards are helpful in representing concepts in cyber security but are more tailored for XML and must be adapted for ontology use.
- Snort based event information in context with NetFlow information can be used to understand the timing, duration and network characteristics of attacks.

Creating an ontology from real networks is non-trivial. However, ontologies and inference can provide additional insightful information beyond path-based analysis about the network. Ontologies provided an easier framework for incorporating new information and rules compared to traditional Rete based rule engines such as Drools. However, the use of ontologies does require domain knowledge, ontology and software development skills to be successful that are not commonly part of network administration.

The cyber ontologies developed here leveraged various standards developed by MITRE, NIST, USCERT and other organizations. Many of the standards were represented as XML, which was found during the research to be very difficult to automatically convert to an ontology using XML translation rules. The lack of OWL ontologies corresponding to the MITRE standards and other cyber situational standards hinder interoperability and situational awareness because the XML schemas do not have a formal semantics. Thus, information in those standards cannot be combined and used to infer new knowledge. The STIX based ontology developed in this project could be used as a starting point by various organizations that are currently in the process of defining the STIX standards using XML as well as other research organizations. Lessons learned on the conversion process of the MITRE and STIX XML would help others to avoid pitfalls. The STIX ontology could also aid in inter agency and department cyber information sharing as it would help add semantic meaning, but due to the XML usage it may be difficult

to get an ontology version that can be kept up to date.

The STIX community has expressed interest in an eventual OWL encoding of its work, but this task is non-trivial. Serious efforts need to begin on constructing interoperable ontologies for cyber situational awareness so that inferences based on shared information can be made in standard, transparent, uniform ways. Since so much valuable work has already gone into developing existing XML-based standards, and each of these has an existing community of users, it makes sense to invest serious effort into developing adequate techniques for lifting XML schemas into OWL ontologies so that machines can share not only common vocabularies, but common meanings for representing and inferring the state of networks.

9. Summary

In cyber systems, where the processing loops are very fast, much of inference must be performed by computers. In other words, automatic inference engines must perform the inference, which in turn requires that the information (facts) to be acted upon by such engines needs to be represented in a language with formal semantics. The term “ontology” stands for an explicit, formal, machine-readable semantic model that defines the classes, instances of the classes, inter-class relations and data properties relevant to a problem domain. In order for ontologies to be amenable to automatic processing by computers, they need to be represented in a language that has both formal syntax and formal semantics. OWL is the most commonly used language for expressing ontologies today and has by far the largest developer base. Automatic inference on ontologies expressed in OWL is performed by inference engines, or semantic reasoners. An inference engine takes a set of facts about a specific domain of interest asserted in OWL and derives other facts using axioms and inference rules. OWL’s semantics is such that inference is expected to be computationally tractable. Several commercial inference engines for making inferences with OWL ontologies are available. The expressive power of OWL is relatively high—equivalent to a decidable fragment of complete First Order Logic. In addition, OWL can be supplemented with rules. There have been relatively few attempts to develop ontology to comprehensively encode cyber situation information as OWL ontology, and no comprehensive ontologies have been published. The most relevant standard is a relatively new effort supported by various organizations and managed by MITRE called Structured Threat Information eXpression (STIX). Other relevant XML schemas exist, however, they merely mandate how information should be structured in conveying a pre-specified set of XML elements from one agent to another. They fall short of being ontology and cannot be subject to automatic inference. Automatic reasoning based on ontologies can support situational awareness in the cyber security domain. An approach that mimics the analyst’s process can be implemented using ontologies and an inference engine. A comprehensive ontology for cyber security can be developed by taking advantage of existing cyber security related standards and markup languages.

References

STIX - Structured Threat Information eXpression. “A Structured Language for Cyber Threat Intelligence Information”. <http://stix.mitre.org>

MAEC - Malware Attribute Enumeration and Characterization. <http://maec.mitre.org/> .

CWE – Common Weakness Enumeration. <http://cwe.mitre.org>

Swimmer, M. Towards An Ontology of Malware Classes. January 27, 2008.
<http://www.scribd.com/doc/24058261/Towards-an-Ontology-of-Malware-Classes>.

[IODEF](http://xml.coverpages.org/iodef.html) - Cover Pages Incident Object Description and Exchange Format (~~IODEF~~).
<http://xml.coverpages.org/iodef.html> .

~~Internet Engineering Task Force. [Online] <http://www.ietf.org/>.~~

CAPEC - Common Attack Pattern Enumeration and Characterization. <http://capec.mitre.org/> .

Cybox - Cyber Observable eXpression. <http://cybox.mitre.org>

jsoup: Java HTML Parser. <http://jsoup.org/>

F. Baader, D. L. McGuinness, D. Nardi and P. F. Patel-Schneider (Eds.). (2010) *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

Apache Jena. <http://jena.apache.org>

WhoIs <http://www.whois.com/>

The Friend of a Friend (FOAF) project. <http://www.foaf-project.org/> .

Hobbs, J. R. and Pan, F. An Ontology of Time for the Semantic Web. CM Transactions on Asian Language Processing (TALIP): Special issue on Temporal Information Processing. 2004. Vol. 3, 1, pp. 66-85.

GeoNames Ontology - Geo Semantic Web. <http://www.geonames.org/ontology/documentation.html> .

NIST. National Vulnerability Database Version 2.2. [http:// http://nvd.nist.gov/](http://nvd.nist.gov/)

[SCAP](http://scap.nist.gov/) - The Security Content Automation Protocol (~~SCAP~~) - NIST. [Online] <http://scap.nist.gov/> .

OVAl - Open Vulnerability and Assessment Language. [Online] <http://oval.mitre.org/> .

CPE - Common Platform Enumeration. [Online] <http://cpe.mitre.org/> .

[CCE](http://cce.mitre.org/) - Common Configuration Enumeration (~~CCE~~): Unique Identifiers for Common System Configuration Issues. [Online] <http://cce.mitre.org/> .

CVE - Common Vulnerabilities and Exposures. [Online] <http://cve.mitre.org/> .

Common Vulnerability Scoring System (CVSS-SIG). [Online] <http://www.first.org/cvss/> .

Parmelee, M. *Toward an Ontology Architecture for Cyber- Security Standards*. George Mason University, Fairfax, VA : Semantic Technologies for Intelligence, Defense, and Security (STIDS) 2010

Obrst, L., Chase, P., & Markeloff, R. (2012). Developing an ontology of the cyber security domain. Proceedings of Semantic Technologies for Intelligence, Defense, and Security (STIDS), 49-56.

A.~~neop~~ Singhal and ~~Duminda~~D. Wijesekera. 2010. Ontologies for modeling enterprise level security

metrics. In *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research* (CSIIRW '10), Frederick T. Sheldon, Stacy Prowell, Robert K. Abercrombie, and Axel Krings (Eds.). ACM, New York, NY, USA, , Article 58 , 3 pages. DOI=10.1145/1852666.1852731 <http://doi.acm.org/10.1145/1852666.1852731>

X. Wang, S. Mamadgi, A. Thekdi, A. Kelliher, and H. Sundaram. Eventory – an event based media repository. In *Semantic Computing*, pages 95–104, Washington, DC, USA, 2007. IEEE. ISBN 0-7695-2997-6.

Y. Raimond and S. Abdallah. The event ontology, October 2007. <http://motools.sf.net/event>

IPTC International Press Telecommunications Council, London, UK. EventML, 2008. <http://iptc.org/> .

M. Doerr, C.-E. Ore, and S. Stead. The CIDOC conceptual reference model: a new standard for knowledge sharing. In *Conceptual modeling*, pages 51–56. Australian Computer Society, Inc., 2007. ISBN 978-1-920682-64-4.

E. T. Mueller. *Handbook of Knowledge Representation*, chapter Event Calculus. Elsevier, 2008

A. R. J. Francois, R. Nevatia, J. Hobbs, and R. C. Bolles. VERL: An ontology framework for representing and annotating video events. *IEEE MultiMedia*, 12(4), 2005.

U. Westermann and R. Jain. Toward a common event model for multimedia applications. *IEEE MultiMedia*, 14(1):19–29, 2007.

A. Scherp, T. Franz, C. Saathoff, and S. Staab. F—a model of events based on the foundational ontology DOLCE+DnS Ultralight. In *Conference on Knowledge Capture*, pages 137–144, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-658- 8. doi: <http://doi.acm.org/10.1145/1597735.1597760>.

H. Chen and A. Joshi. *The SOUPA Ontology for Pervasive Computing*. Birkhauser Publishing Ltd., April 2004

X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung. Ontology based context modeling and reasoning using OWL. In *Pervasive Computing and Communications Workshops*, page 18, Washington, DC, USA, 2004. IEEE. ISBN 0-7695-2106-1

S. S. Yau and J. Liu. Hierarchical situation modeling and reasoning for pervasive computing. In *Software Technologies for Future Embedded and Ubiquitous Systems*, pages 5–10, Washington, DC, USA, 2006. IEEE. ISBN 0-7695-2560-1.

C. J. Matheus, M. M. Kokar, K. Baclawski, and J. Letkowski. An application of semantic web technologies to situation awareness. In *International Semantic Web Conference*, volume 3729 of LNCS, pages 944–958. Springer, 2005.

C. J. Matheus, M. M. Kokar, and K. Baclawski. A core ontology for situation awareness; Cairns, Australia. In *Information Fusion*, pages 545–552, July 2003.

M. M. Kokar, C. J. Matheus, and K. Baclawski. Ontology-based situation awareness. *Inf. Fusion*,

10(1):83–98, 2009. ISSN 1566-2535. doi: <http://dx.doi.org/10.1016/j.inffus.2007.01.004>.

C. Matheus, K. Baclawski and M. Kokar. (2006). BaseVISor: A Triples-Based Inference Engine Outfitted to Process RuleML and R-Entailment Rules. In Proceedings of the 2nd International Conference on Rules and Rule Languages for the Semantic Web, Athens, GA.

F. Lin. *Handbook of Knowledge Representation*, chapter Situation Calculus. El- sevier, 2008

J. Barwise, J. Perry. (1983) Situations and Attitudes. Cambridge, MA: MIT Press.

Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2011). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1, 80.

~~Joe~~J. Stewart, (2013). Chasing APT. Dell SecureWorks Counter Threat Unit™ Threat Intelligence. 23 July 2012. http://www.secureworks.com/research/threats/chasing_ap/
AQ7

US-CERT. (2013) Alert (TA13-309A) CryptoLocker Ransomware Infections. Original release date: November 05, 2013 | Last revised: November 18, 2013 <http://www.us-cert.gov/ncas/alerts/TA13-309A>

I. Bedini et al. “Transforming XML Schema to OWL Using Patterns”. In: Semantic Computing (ICSC), 2011a Fifth IEEE International Conference on. 2011, pp. 102–109. doi: 10.1109/ICSC.2011.77

~~Nikos~~N. Bikakis et al. “The XML and Semantic Web Worlds: Technologies, Interoperability and Integration: A Survey of the State of the Art”. In: Semantic Hyper/Multimedia Adaptation. Ed. by Ioannis

E. Anagnostopoulos et al. Vol. 418. Studies in Computational Intelligence. Springer Berlin Heidelberg, 2013, pp. 319–360. isbn: 978-3-642-28976-7. doi: 10 . 1007 / 978 - 3 - 642 - 28977 - 4 _ 12. url: http://dx.doi.org/10.1007/978-3-642-28977-4_12

~~Matthias~~M. Ferdinand, ~~Christian~~C. Zirpins, and ~~David~~D. Trastour. “Lifting XML Schema to OWL”. In: Web Engineering. Ed. by Nora Koch, Piero Fraternali, and Martin Wirsing. Vol. 3140. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 354–358. isbn: 978-3-540-22511-9. doi: 10.1007/978-3-540-27834-4_44. http://dx.doi.org/10.1007/978-3-540-27834-4_44.

~~Hannes~~H. Bohring and ~~Sören~~S. Auer. “Mapping XML to OWL Ontologies.” In: Leipziger Informatik- Tage 72 (2005), pp. 147–156.

~~Steve Bratt. Semantic Web and Other Technologies to Watch. Slide 24.~~
~~<http://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/24>~~

Michel Dumontier. SemanticScience wiki: ODPMereotopology.
<https://code.google.com/p/semanticsscience/wiki/ODPMereotopology> . Updated Nov 27, 2013.

~~Toni~~T. Rodrigues, ~~Pedro~~P. Rosa, and ~~Jorge~~J. Cardoso. “Mapping XML to Existing OWL ontologies”. In: International Conference WWW/Internet. Citeseer. 2006, pp. 72–77.

Janez J. Brank, Marko M. Grobelnik, and Dunja D. Mladenic. A survey of ontology evaluation techniques. In In Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005)

Leo L. Obrst, Werner W. Ceusters, Inderjeet I. Mani, Steve S. Ray, and Barry B. Smith. The evaluation of ontologies. In Christopher J.O. Baker and Kei-Hoi Cheung, editors, Semantic Web, pages 139{158. Springer US, 2007.

Zeqian Z. Shen, K-L Ma, and Tina T. Eliassi-Rad. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. Visualization and Computer Graphics, IEEE Transactions on, 12(6):1427{1439, 2006.

Denny D. Vrandečić. Ontology evaluation. In Stephen Staab and Rudi Studer, editors, Handbook on Ontologies, International Handbooks on Information Systems, pages 293{313. Springer Berlin Heidelberg, 2009.

Juan J. Ye, Lorean L. Coyle, Simon S. Dobson, and Paddy P. Nixon. Ontology-based models in pervasive computing systems. The Knowledge Engineering Review, 22(4):315{347, 2007.

J. Boyd. A discourse on winning and losing. Technical report, Maxwell AFB, 1987.

M. Endsley, Mica (1995). “Toward a theory of situation awareness in dynamic systems”. Human Factors 37(1), 32-64.

T. Gruber. Ontology. In Ling Liu and M. Tamer Ozsu, editors, The Encyclopedia of Database Systems, pages 1963–1965. Springer, 2009.

W3C. OWL 2 Web Ontology Language Document Overview, 2009. <http://www.w3.org/TR/owl2-overview/> .

OWL/Implementations. W3C. <http://www.w3.org/2001/sw/wiki/OWL/Implementations> .

RDF: Resource Description Framework. W3C. <http://www.w3.org/RDF/>

I. Horrocks and U. Sattler. The effect of adding complex role inclusion axioms in description logics. In Proc. of the 18th Int. Joint Conf. on Artificial Intelligence (IJCAI 2003), pages 343–348. Morgan Kaufmann, Los Altos, 2003.

Ivan I. Bedini, Christopher C. Matheus, Peter P. F. Patel-Schneider and Aidan A. Boran. Transforming XML Schema to OWL Using Patterns. ICSC '11 Proceedings of the 2011 IEEE Fifth International Conference on Semantic Computing, Pages 102-109, 2011b.

Ashwini A. D. Khairkar, Deepak D. Kshirsagar and Sandeep S. Kumar. “Ontology for Detection of Web Attacks”, International Conference on Communication Systems and Network Technologies, 2013.

Wentao W. Kang and Ying Y. Liang. “A Security Ontology with MDA for Software Development”, International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, 2013.

D. McMorow. Science of Cyber-Security. Technical Report, JSR-10-102, The MITRE Corporation,

2010.

David D. A. Mundie and **David D.** M. McIntire. "The MAL: A Malware Analysis Lexicon", Technical Note, CMU/SEI-2013-TN-010, Software Engineering Institute, 2013.

Anita A. D'Amico, **Laurin L.** Buchanan, **John J.** Goodall and **Paul P.** Walczak. "Mission Impact of Cyber Events: Scenarios and Ontology to Express the Relationships between Cyber Assets, Missions and Users", International Conference on i-Warfare and Security (ICIW), The Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, USA, 2010.

John J. R. Goodall, **Anita A.** D'Amico and **Jason J.** K. Kopylec. "Camus: Automatically Mapping Cyber Assets to Missions and Users", IEEE Military Communications Conference, MILCOM 2009, pp.1-7, 2009.

Henry H. M. Kim, **Markus M.** Biehl and **John J.** A. Buzacott. "M-CI²: Modelling Cyber Interdependencies between Critical Infrastructures", 3rd IEEE International Conference on Industrial Informatics (INDIN), 2005.

J. Undercoffer, A. Joshi, and J. Pinkston, "Modeling Computer Attacks: An Ontology for Intrusion Detection," in Proc. 6th Int. Symposium on Recent Advances in Intrusion Detection . Springer, September 2003.

Chris C. Strasburg, **Samik S.** Basu and **Johnny J.** S. Wong. "S-MAIDS: A Semantic Model for Automated Tuning, Correlation, and Response Selection in Intrusion Detection Systems", IEEE 37th Annual Computer Software and Applications Conference, 2013.

Arwa A. Wali, **Soon Ae S.** A. Chun and **James J.** Geller. "A Bootstrapping Approach for Developing a Cyber-Security Ontology Using Textbook Index Terms", International Conference on Availability, Reliability and Security, 2013.

A. Herzog, N. Shahmehri, and C. Duma, "An Ontology of Information Security," IGI Global, 2007, pp. 1-23.

Mathieu M. Bouet and **Maurice M.** Israel. "INSPIRE Ontology Handler: automatically building and managing a knowledge base for Critical Information Infrastructure Protection", 12th IFIP/IEEE IM, 2011.

Giuseppe G. Fenza, **Domenico D.** Furno, **Vincenzo V.** Loia, and **Mario M.** Veniero. "Agent-based Cognitive approach to Airport Security Situation Awareness", International Conference on Complex, Intelligent and Software Intensive Systems, 2010.

John J. Strassner, **Joseph J.** Betser, **Roberta R.** Ewart and **Frank F.** Belz. "A Semantic Architecture for Enhanced Cyber Situational Awareness", Secure& Resilient Cyber Architectures Conference, MITRE, McLean, VA, 2010.

Jeffrey J. M. Bradshaw, **Marcos M.** Carvalho, **Larry L.** Bunch, **Tom T.** Eskridge, **Paul P.** J. Feltovich, **Matt M.** Johnson and **Dan D.** Kidwell. "Sol: An Agent-Based Framework for Cyber Situation Awareness", Kunstl Intell, 26:127-140, 2012.

Alessandro A. Oltramari, **Christian C.** Lebiere, **Lowell L.** Vizenor, **Wen W.** Zhu and **Randall R.** Dipert.

“Towards a Cognitive System for Decision Support in Cyber Operations”, STIDS, 2013.

~~Sumit~~S. More, ~~Mary~~M. Matthews, ~~Anupam~~A. Joshi, ~~Tim~~T. Finin. “A Knowledge-Based Approach To Intrusion Detection Modeling”, IEEE Symposium on Security and Privacy Workshops, 2012.

~~James~~J. S. Okolica, ~~J. Todd~~T. McDonald, ~~Gilbert~~G. L. Peterson, ~~Robert~~R. F. Mills, and ~~Michael~~M. W. Haas. Developing Systems for Cyber Situational Awareness. Proceedings of the 2nd Cyberspace Research Workshop, Shreveport, Louisiana, USA, 2009.

Alexandre de Barros Barreto, Paulo Cesar G Costa and Edgar Toshiro Yano. Using a Semantic Approach to Cyber Impact Assessment. STIDS, 2013.

~~Amit~~A. Sheth, Can Semantic Web techniques empower comprehension and projection in Cyber Situational Awareness? ARO Workshop, Fairfax, VA, 2007.

~~Jeffrey~~J. L. Caton. “Beyond Domains, Beyond Commons: Context and Theory of Conflict in Cyberspace”, 4th International Conference on Cyber Conflict, 2012.

~~Simon Reay~~S. R. Atkinson, ~~Kevin~~K. Beaulne, ~~David~~D. Walker and ~~Liaquat~~L. Hossain. “Cyber – Transparencies, Assurance and Deterrence”, International Conference on Cyber Security, 2012.

CWE - Common Weakness Enumeration. National Vulnerability Database, <http://nvd.nist.gov/cwe.cfm>.

Security Intelligence: Defining APT Campaigns. SANS Digital Forensics and Incident Response, <http://digital-forensics.sans.org/blog/2010/06/21/security-intelligence-knowing-enemy/>

AQ5

AQ6