Experiences in Using Semantic Reasoners to Evaluate Security of Cyber Physical Systems

Bruce Barnett, Andrew Crapo, Paul O'Neil

GE Global Research Center
1 Research Center, Schenectady NY 12309

barnettbr@ge.com, crapo@ge.com, oneilp@ge.com

Keywords: Cyber Security, Smart Grid, Ontology, Semantic Web

Abstract

Smart grid security is challenging as experts in both IT Security and Industrial Control Systems (ICS) are few. Expertise in multiple domains is needed, and tools that can be used to analyze smart grid systems during the design phase are non-existent. We used Semantic Web Technology to create an ontology that is capable of reasoning about security attributes. The ontology model and rules were written in Semantic Application Design Language (SADL). Input was divided into reusable knowledge (ontology, rules, device specifications), and site-specific information (topology, configuration options). SADL uses English-like statements to build a model that is understandable by domain experts without requiring knowledge of Semantic Web Technology.

In this discussion, we present components of the ontology that are capable of describing, measuring, and comparing both physical and network-specific risks. We also developed a GUI that displayed the results in a Failure Mode Effects Analysis, where threats are prioritized by Likelihood, Detectability, and Severity.

1. INTRODUCTION

Addressing the needs for security in cyber systems is difficult for several reasons. These difficulties are outlined in the sections below.

1.1. Multiple domain experts needed

When considering the Smart Grid and Supervisory Control and Data Acquisition (SCADA)/ICS (Industrial Control Systems) finding experts is problematic.

Security is a difficult problem in any domain, as it requires expert knowledge from multiple domains—more than any single Subject Matter Expert can master. To elaborate, security expertise can be subdivided into several categories including:

- Physical security (buildings, fences, locks, alarms)
- Device security (requires specialized knowledge of the device)
- Reverse engineering (ability to extract and duplicate proprietary algorithms and code from compiled programs)
- Network control security (requires knowledge of network topology and protections)
- Application security (requires knowledge of the SCADA software accessing a device)
- Network protocol security (requires understanding of protocols and packet contents)
- Operational security (requires knowledge of the overall system requirements and functionality)

These areas tend to be specialized. Any unifying framework would need to integrate the information necessary for different security domains.

1.2. Non-formal security metrics

Reviews of system architecture are typically performed by a panel of experts in the areas above. However, as each expert has a unique background, they bring different perspectives to the task. Results depend on the quality of the experts. If such experts are not available, then the review may be incomplete. Different experts may generate different results. If a more formal and mathematical model were used, then analysis would be consistent and automated.

While some formal models have been created, their purposes are specialized, and not suitable for the smart grid. One example is OSATE, based on AADL (Architecture Analysis and Design Language) [1],[2], which provides binding based on constraint-based resource allocation. OSATE addresses Bell-LaPadula, Biba, and MILS models, but Smart Grid systems don't need multi-level security. Another formal model, SecOntManager [3], deals with IT-Security. SecOntManager has been used to analyze loss of assets caused by fire damage, and others aspects of IT security. Lemay et al., have developed the ADVISE model

[4] suitable for the Smart Grid. However, the model, rules, and architecture are integrated. We believe a model where the rules are independent from the topology would be easier to deploy.

1.3. Our Approach

Donner [5] discusses the need for a security ontology to "report incidents more effectively, share data and information across organizations, and discuss issues among ourselves". If a formal ontology was used, then tools could be applied. Ideally, these tools should have the following capabilities:

- Ability to measure system security in an objective and repeatable fashion.
- Ability to merge information from multiple domains of knowledge.
- Ability to capture and re-use knowledge and expertise from experts.

We proceeded to build a prototype to determine the suitability of the technology using semantic reasoners.

2. WHAT IS THE SEMANTIC WEB?

Communication will always be imprecise when there is disagreement over the meaning of words. The word "network", for instance, has different meanings to companies such as NBC, Cisco, and Facebook. A formal definition of terms eliminates ambiguity.

The World Wide Web provides syntactically correct information to a web browser, which has the task of rendering the data. However, complex problems either require humans to interpret the data, or specially crafted scanners to interpret web pages. Knowledge is difficult to capture and reuse.

The Semantic Web, a term coined by Tim Berners-Lee [6], is a technology that (a) defines an ontology, (b) provides meaningful data using this ontology over the web, (c) provides rules that use this data, and (d) allows semantic reasoners to deduce information using the rules and data.

3. SEMANTIC WEB STANDARDS

The Web Ontology Language, [7], is a formal (unambiguous) language that builds on RDF and RDFS and is informed by prior work including Description Logics (DL), OIL, DAML, and DAML+OIL. The Semantic Web Rule Language [8] is a W3C proposal that combines OWL (DL and Lite) with a subset of the Rule Markup Language (RuleML). Alternately, Jena [9] which understands the Jena rule language, or Pellet [10] which uses SWRL, may be used. The results can be queried using the SPARQL RDF query language [11].

However, developing tools based on OWL, SWRL, and SPARQL is difficult. OWL is designed for computer parsing systems, not for easy consumption by people. Similarly SWRL syntax is not English-like and is difficult to understand. If an ontology is modified in an OWL file, the corresponding change in a SPARQL query is not obvious and may not be immediately identified. Discovery of a malformed query occurs when the query fails, or when looking for inconsistencies across several files in different hard-to-comprehend dialects.

4. USING SADL TO DEVELOP SEMANTIC WEB APPLICATIONS

SADL [12] is an open-source solution developed by GE that addresses the above problems. It allows ontologies to be expressed in an English-like language. Ontologies written in SADL are automatically converted into OWL. Rules written in SADL can be converted to Jena rules or SWRL, allowing the user in the Eclipse environment to launch and query semantic reasoners, such as Jena and Pellet. Ontology modifications that break existing rules and models are immediately identified while entering text using the Eclipse-based SADL development environment, and can be addressed in the editor, without executing any queries. In other words, rules and ontologies can be easily modified and reasoning solutions can be quickly developed. SADL color-codes the different components based on their type (Class, Instance, Attribute, and Literal), uses red to indicate a syntax error, and marks formal modeling errors with a flag.

The use of SADL to model Smart Grid systems has been previously demonstrated [13],[14]. We decided to apply SADL technology to cyber security issues for a Smart Grid system.

5. SEPARATION OF KNOWLEDGE DOMAINS

We created an ontology that separated certain domains, allowing each to provide input independently. These domains include:

- Device library – allowing the specification of device characteristics independent from configuration-specific information. Each device type is a unique class.
- Physical topology – allowing the generation of network topology information from a network layout design package.
- Network protection – specifying the controls that protect and isolate a network, using information provided by the network operations department

We divided the threats into physical and network-related threats to test the usefulness of the technology.

6. PHYSICAL SECURITY

We created the following ontology to describe attributes necessary to calculate both physical and cyber security. The SADL specification is shown in Figure 2.

Our base class is **Device**. To provide a means to analyze physical security, we created a generalization of the base class called **Container.** Containers are used to describe physical barriers (fences, buildings) as well as electronic cabinets, racks, and cases used to house electronics. Any device may have the ability to detect tampering in the model. As an example, the layers of protection for a power relay and the values of the protection attributes are summarized in Table 1. In our ontology, the outermost containers can be identified because they are not contained inside of other containers. We created rules to identify these items and calculate the total hours to penetrate all of the physical defenses and expose the innermost device. Using the data in Table 1, the total effort would be 17 hours. This value can be divided by a variable that classifies the skill of the attacker to estimate the elapsed time of a physical compromise. For example, an Expert (value of 2.0) could breech the physical defenses in 8.5 hours and a Novice (value of 0.5) would take 34 hours.

To calculate the probability of detection, each of the physical barriers optionally had a tamper detection probability. We used formula (1) to calculate the total Detection Probability (DP). In the example in Table 1, the chassis and the building have alarms, with probability of the alarm detecting tampering at 80% and 90% respectively.

Using the data in Table 1, the overall probability of detection would be 1- (1-.90)*(1-.80) = .98 (98%). We did not modify the detection probability based on the skill of the attacker. Better models for attacker skillsets are, or course, possible.

We added a **numberOfHomesServicing** attribute that specified the number of homes that would be impacted if the device was compromised. The **controlOf** attribute was used to describe when one device has supervisory control over another device. The cumulative number of homes under control of a single device could then be calculated using rules that total up the number of homes under the control of a single device, both directly and indirectly (when a compromised device has control of a non-compromised device).

$$\sum DP = 1 - (1-DP(\text{Layer}_1) * (1-DP(\text{Layer}_2)) \ldots * (1-DP(\text{Layer}_n)) \quad (1)$$

Table 1 - Example of Physical Protection by Containers

Table of Physical Protection Characteristics		
Device	Physical Penetration Effort (hours)	Tamper Detection Probability
Relay Electronics	2	0
Relay Case	1	0
Chassis	1	80%
Building	10	90%
Fence	3	0

.

These specified and derived attributes allow us to calculate the detectability, likelihood, and severity of a physical compromise for any device using a modified Failure Mode Effects Analysis (FMEA) formula. We wrote a Java client that queries the reasoning engine and extracts this information. To allow comparison, we normalized the raw values, converting them to a value from 1 to 10, using a scale appropriate for each measurement. A higher number corresponded to a greater impact of the device's compromise. The three values are multiplied to create a Risk Prioritization Number (RPN). By sorting on the RPN, we could identify devices that have the greatest impact if compromised. A screenshot of this tool is shown in Figure 2.

7. NETWORK CONNECTIVITY

We needed a mechanism to determine network connectivity. We first created a class corresponding to Layer 1 of the OSI mode, i.e. **PhysicalLinkLayer**. We also created two generalizations of the class Device. **ComDevice** describes a **Device** with a single network connection. **ControllingDevice** describes a device

that connects two or more networks together, providing connectivity between two or more **PhysicalLinkLayer** networks.

The two Boolean attributes, **providesMACBridging** and **providesIPRouting** are used to describe switches, hubs, routers, and firewalls. We wrote rules that derive MAC-level and IP-Level connectivity based on these attributes, allowing us to derive attributes of instances of the class **Device**. These attributes, **hasMACAccessTo** and **hasIPAccessTo**, allow us to test for connectivity between any two.

8. NETWORK- BASED ATTACKS

Subsequently we created a generalized attack/defense ontology (Figure 1) that can manage complicated and multiple attacks. The ontology describes device capabilities, attack mechanisms, vulnerabilities, and defense mechanisms as classes, and the specific attack or defense is modeled as an instance of the class. This has the benefit of allowing new vulnerabilities to be added without requiring new rules to be written.

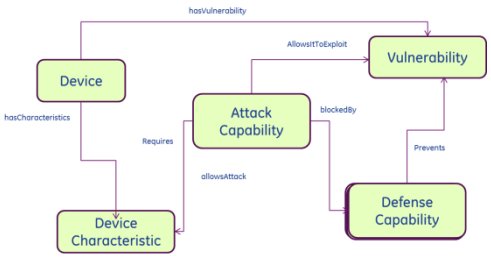


Figure 1 - Attack/Defense Ontology

Before an attack is possible, the device must have Device Characteristics that allow the attack (in addition to being compromised). These characteristics could include required operating systems, the ability to promiscuously sniff traffic, the ability to have raw socket access allowing packets to be written with false information (such as modified MAC or IP addresses), current patch level, access privileges, etc. A compromised device with these characteristics can attack another device using a specific Attack Capability. Each attack is associated with a specific vulnerability associated with a device, and the target of the attack must have the vulnerability.

For each attack, there may or may not be a protection mechanism available. This could be host-based or network-based. If the vulnerable host has the defensive capability, it can prevent that specific attack. If not, it could become compromised.

The defense capability can be network-based as well. For example, a firewall could protect a device from attack, but only if that device is between a compromised device and the target.

We created rules that determined if any **ControllingDevice** had the defensive capability for a particular vulnerability, and was also between the attacker and the target. If so, the target would not become compromised. Therefore the model is capable of determining which vulnerabilities pose a danger of compromise, resulting in the compromise of additional devices, without the need of writing new rules for each vulnerability of the same type.

Using a SADL-based prototype based on this ontology, we created an instance of a StuxNet-like virus. The modeling environment determined which devices lacked protection, and were therefore vulnerable. This allowed us to perform "what-if" analysis, and by calculating the total **numberOfHomesServicing** for all devices either directly compromised, or under control of a compromised device, we were able to calculate the Defense-in-Depth for each device in the network. That is, we determined how machines might become compromised (either directly, or if a compromised machine controls them indirectly).

We designed and verified the model using SADL within Eclipse. As this environment has no graphics capability, we installed the model into a knowledge server running on Apache's Tomcat web server [**Error! Reference source not found.**]. We built a Python-based GUI to query the knowledge server. It is able to draw a diagram of the network connectivity, allow the user to specify which device is compromised, and then indicate the number of devices that are now vulnerable by both network attack and by being under the control of a compromised device.

9. OTHER APPLICATIONS

We have created rules to associate clusters of networks into labeled zones. The knowledge of this association can propagate across networks linked by Layer 2 and 3 devices that do not provide firewall capability. Inconsistent zone assignment can be detected. Devices that connect two or more zones together can be checked to see if they have an Intrusion Detection system attached, as per NERC/CIP requirements [16].

10. EXPERIENCES USING SEMANTIC REASONERS

While applying this technology to analyze cyber-security metrics, we learned of some shortcomings, and some advantages.

- **Semantic Reasoners are not procedural**

Semantic Reasoning is not a procedural programming paradigm. All OWL information is stored as triples of the form <Name, AttributeName, Value>. Because semantic reasoners often require monotonicity, "changing" the value of an input attribute may have unexpected consequences. Even if an input is changed, the conclusions are not automatically changed. To consider other input parameters, the model must be cleared and then the input conditions changed, causing new conclusions to be reached.

Unlike procedural languages and databases, OWL uses the Open World Assumption [17], meaning that not knowing that something is true is not the same as knowing that it is false. For example, a Boolean condition can be true, false, or unknown. To simplify our rules, we emulated the Closed World model by only setting Booleans to True and interpreting the absence of a True value as implying the negative condition.

Creating rules for cyber security requires learning new paradigms of programming. There was a learning curve.

- **Rules are formalized, and limited**

The rule languages are formally defined, and fixed. For example, there are no "*else"* constructs in a rule, and no way to include an "*or"* condition. Instead, new rules were created with opposite preconditions, to reach opposite conclusions. An "or" is implemented as two separate rules capturing the two conditions. We used naming conventions to keep related rules together.

- **Semantic reasoners are not like discrete event simulators**

Unlike a discrete-event simulation, a semantic reasoner has no clock, event list, random number generator, or ending condition. There are no "do-loops" or "while-loops." As long as knowledge can be inferred, the rules will fire. When no further knowledge can be determined, the "simulation" stops.

- **Execution is not ordered**

Rules are "fired" when their premises are satisfied. There is no control of order of firing of the rules. Control of the simulation is managed by importing different rules and attributes for different experimental conditions. Having a systematic mechanism for managing rules and configurations is helpful. We also found the use of test suites valuable to verify the functionality of certain components of the model (like network connectivity).

- **Ontology development is crucial**

The most difficult task is developing an ontology that provides flexibility and extensibility. We invite others to use, modify and extend our current model.

11. SUMMARY AND CONCLUSIONS

We believe that the two prototypes we built demonstrate that semantic reasoners can provide useful benefit in measuring the overall security of a complex network. We met the desired goals with the described technology by (a) measuring the security of a complex system in both an objective and repeatable way, (b) merging information from multiple domains of expertise, and (c) being able to capture and reuse the knowledge of experts.

When developing ontology-based solutions, SADL provides both flexibility and a rapid prototyping environment for developing new ontologies. In our ontological model, only a few attributes and classes are needed to calculate metrics related to physical and network security.

The ontology presented is simple and practical, and can be extended to more sophisticated applications. It allows expression of information from multiple domains, and therefore can be used to combine the expertise of multiple experts. In addition, knowledge can be incrementally added to this model, improving the sophistication and capability of the model as needed. The proper ontology provides a framework for multiple applications to share and exchange information.

References

1. Hansson, J; Lewis, B; Hugues, J; Wrage, L; Feiler, P; Morley, J; , "Model-Based Verification of Security and Non-Functional Behavior using AADL," *Security & Privacy, IEEE* , vol.PP, no.99, pp.1, 0
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5306062&isnumber=5210089
2. Architectural Analysis & Design Language, URL: http://www.aadl.info/
3. Andreas Ekelhart; Stefan Fenz; Markus Klemen; Edgar Weippl; , "Security Ontologies: Improving Quantitative Risk Analysis," *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on System Sciences*, vol., no., pp.156a, Jan. 2007
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4076691&isnumber=4076362
4. LeMay, E.; Ford, M.D.; Keefe, K.; Sanders, W.H.; Muehrcke, C.; , "Model-based Security Metrics Using ADversary VIew Security Evaluation

(ADVISE)," *Quantitative Evaluation of Systems (QEST), 2011 Eighth International Conference on* , vol., no., pp.191-200, 5-8 Sept. 2011 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6042046&isnumber=6041591

5. Donner, M. "Towards a Security Ontology", IEEE Security and Privacy, 1(3). 6-7. URL: http://nygeek.files.wordpress.com/2010/01/j3006.pdf
6. Berners-Lee, T. "The Semantic Web", Scientific American, May 2001, URL: http://www.scientificamerican.com/article.cfm?id=the-semantic-web
7. "OWL Web Ontology Language Reference", February 10, 2004, Mike Dean and Guus Schreiber, eds. http://www.w3.org/TR/owl-ref/
8. "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", May 21, 2004, Ian Horrocks et. al., URL: http://www.w3.org/Submission/SWRL/
9. Jena – A Semantic Web Framework for Java. URL: http://jena.sourceforge.net/
10. Pellet: OWL 2 Reasoner for Java. URL: http://clarkparsia.com/pellet/
11. SPARQL Query Language for RDF. URL: http://www.w3.org/TR/rdf-sparql-query/
12. SADL SourceForge Home URL: http://sadl.sourceforge.net/
13. Andrew Crapo, Xiaofeng Wang, John Lizzi, and Ron Larson. "The Semantically Enabled Smart Grid" (2009). Grid Interop 2009. URL: http://www.gridwiseac.org/pdfs/forum_papers09/crapo.pdf
14. Andrew Crapo, Katrina Griffith, Ankesh Khandelwal, John Lizzi, Abha Moitra, Xiaofeng Wang, "Overcoming Challenges Using the CIM as a Semantic Model for Energy Applications", Grid-Interop 2010. URL: http://www.pointview.com/data/files/3/2433/1730.pdf
15. http://tomcat.apache.org/
16. Standard CIP–005–1 — Cyber Security — Electronic Security Perimeter(s), R1.5 URL: http://www.nerc.com/files/CIP-005-1.pdf
17. Open World Assumption, Wikipedia, URL: http://en.wikipedia.org/wiki/Open_world_assumption
18. [LEM11] Elizabeth LeMay, Michael D Ford, Ken Keefe, William H. Sanders, "Model-based Security Metrics using ADversary VIew Security Evaluation (ADVISE)", https://www.perform.csl.illinois.edu/Papers/USAN_papers/11LEM01.pdf
19. "Cyber Security Job Outlook" http://cybersecuritydegreeprograms.com/cyber-security-news/cyber-security-job-outlook/
20. "Cyber Security jobs survey", https://cybersecuritychallenge.org.uk/jobs-survey.php, Captured Feb 2012
21. MITRE, "Making Security Measurable", URL: http://measurablesecurity.mitre.org/
22. Fenz, S., "Ontology-based generation of IT-security metrics", SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing
23.

Biography

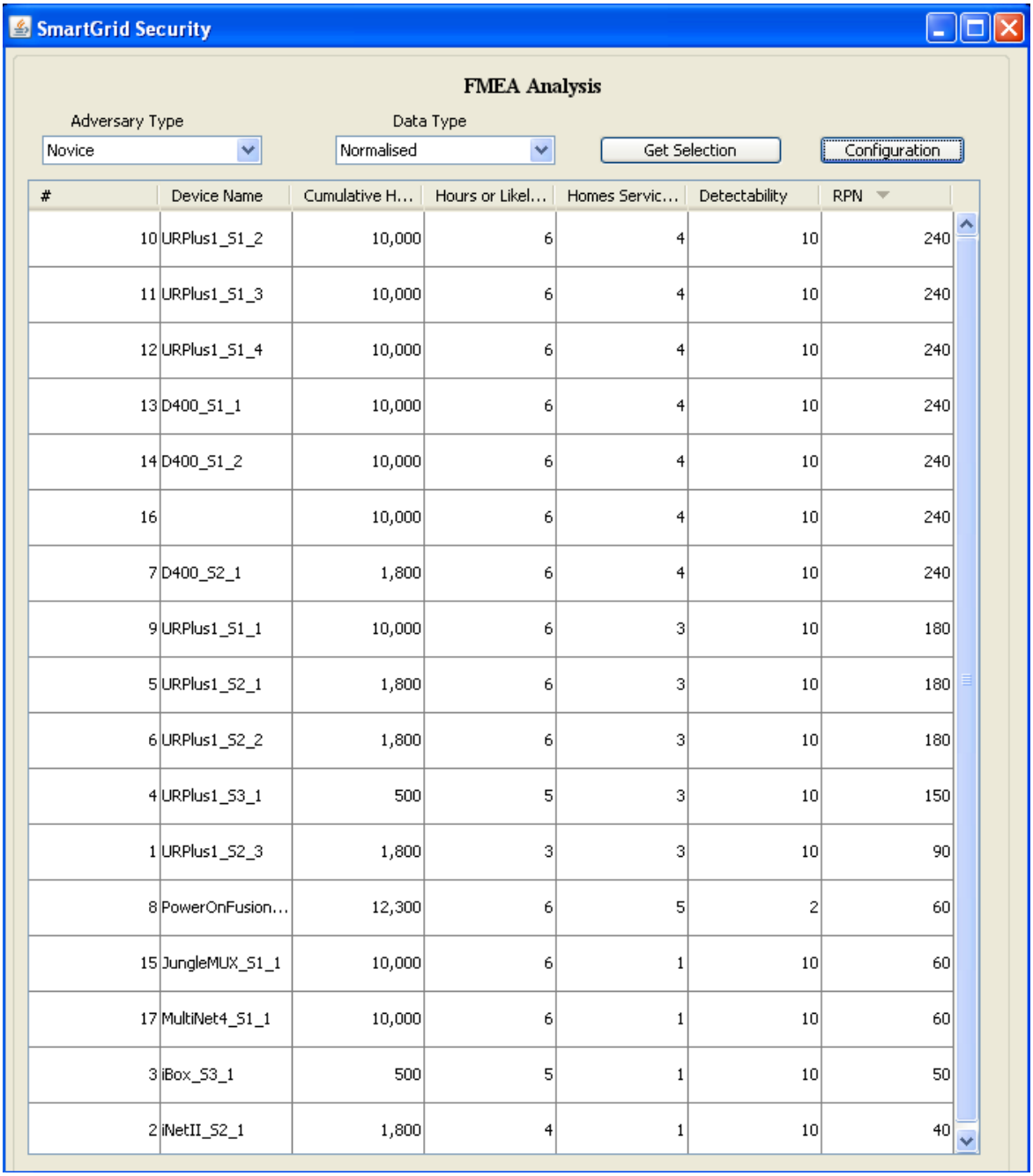
Bruce Barnett graduated with a BS in Mathematics of Computation from RPI in 1973. Bruce was the primary Software Engineer and for 12 years developed and maintained the major product line of the Factron division of Schlumberger. In 1988, Bruce joined GE's Global Research Center in Niskayuna, NY. Bruce developed working prototypes and published 21 papers on secure wireless sensor protocols (2005), complexity-based intrusion detection system (2002-2009), secure data provenance (2009), security vulnerability (2000) and expert-system-based fault analysis programs (1995) using GEN-X with Andy.

Dr. Andrew Crapo received a B.S. in Physics from Brigham Young University in 1975, an M.S. in Energy Systems from the University of Central Florida in 1980, and a Ph.D. in Decision Sciences and Engineering Systems from Rensselaer Polytechnic Institute in 2002. He is a senior professional information scientist at the GE Global Research Center where he has worked since 1980. His work has focused on applications of information science to engineering problems including applied artificial intelligence, human-computer interactions, and information system architectures. More recently he has focused on

modeling and the application of Semantic Web technologies to engineering and business problems.

Paul O'Neil

Paul has a MS in Psychology (Capella University) and a MS in Information Assurance (Norwich University). He has 10 years of computer network and software security experience. Leads experimentation related to cyber security related technology that includes literature searches, modeling and simulation, prototype design, experimentation, and results analysis. He has experience SIEM and data loss technology.



#	Device Name	Cumulative H...	Hours or Likel...	Homes Servic...	Detectability	RPN
10	URPlus1_S1_2	10,000	6	4	10	240
11	URPlus1_S1_3	10,000	6	4	10	240
12	URPlus1_S1_4	10,000	6	4	10	240
13	D400_S1_1	10,000	6	4	10	240
14	D400_S1_2	10,000	6	4	10	240
16		10,000	6	4	10	240
7	D400_S2_1	1,800	6	4	10	240
9	URPlus1_S1_1	10,000	6	3	10	180
5	URPlus1_S2_1	1,800	6	3	10	180
6	URPlus1_S2_2	1,800	6	3	10	180
4	URPlus1_S3_1	500	5	3	10	150
1	URPlus1_S2_3	1,800	3	3	10	90
8	PowerOnFusion...	12,300	6	5	2	60
15	JungleMUX_S1_1	10,000	6	1	10	60
17	MultiNet4_S1_1	10,000	6	1	10	60
3	iBox_S3_1	500	5	1	10	50
2	iNetII_S2_1	1,800	4	1	10	40

Figure 2 -Screen Capture of FMEA tool

```
Vulnerability is a top-level class.

OSILayer is a top-level class.

PhysicalLinkLayer is a type of OSILayer.

Device is a top-level class, // Base device class
described by controlOf with values of type Device,
described by physicalPenetrationEffort has a single value of type float,
described by tamperDetectionProbability with a single value of type float,
described by numberOfHomesServicing with a single value of type int,
described by hasCharacteristics with values of type DeviceCharacteristic.

ComDevice is a type of Device, // with a single network interface
described by attachedTo with values of type PhysicalLinkLayer,
described by hasMACAccessTo with values of type ComDevice,
described by hasIPAccessTo with values of type ComDevice,
described by hostDefenseCapabilities with values of type HostDefenseCapability.

ControllingDevice is a type of ComDevice, // with 2 or more network interfaces
described by providesMACBridging with a single value of type boolean,
described by providesIPRouting with a single value of type boolean
described by networkDefenseCapabilities with values of type NetworkDefenseCapability.

Container is a type of Device,
described by containsDirectly with values of type Device.

DeviceCharacteristic is a top-level class,
described by allowsAttack with values of type AttackCapability.

DefenseCapability is a top-level class,
described by prevents with values of type AttackCapability.

NetworkDefenseCapability is a type of DefenseCapability.

HostDefenseCapability is a type of DefenseCapability.

AttackCapability is a top-level class,
described by blockedBy with values of type DefenseCapability,
described by allowsItToExploit with a single value of type Vulnerability,
described by requiresCharacteristic with values of type DeviceCharacteristic.
```

Figure 3- SADL Ontology