An Ontological Model for Constructing Möbius ADVISE Security Models

Brett Feddersen, Ken Keefe, William H. Sanders
Information Trust Institute
University of Illinois
Urbana, Illinois, USA
{bfeddrsn, kjkeefe, whs}@illinois.edu

Carol Muehrcke, Donald Parks
Cyber Defense Agency
Wisconsin Rapids, Wisconsin, USA
{cmuehrcke, cparks}@cyberdefenseagency.com

Andrew Crapo, Alfredo Gabaldon, Ravi Palla
General Electric Global Research
Niskayuna, New York, USA
{crapo, alfredo.gabaldon, palla}@ge.com

***Abstract*—The ADVISE modeling formalism provides a quantifiable, executable model of adversary behavior for studying the security of complex systems. Each ADVISE model consists of an attack execution graph that defines all attack steps that can be performed by an adversary and how the performance of those attack steps changes the system, as well as an adversary profile that specifies the interests and capabilities of an adversary, which drives the path the attacker chooses to pursue. In order to greatly simplify and improve modeling with the ADVISE formalism, we propose an ADVISE meta model for specifying a system, a set of adversaries, and a set of metrics that can be used to generate an ADVISE model containing a complete set of attack steps for the system. This paper introduces Web Ontology Language libraries, which are used to build ADVISE meta models and automatically generate attack execution graphs.**

I. Introduction

Creating secure complex systems of networks, computers, and humans is a difficult, but necessary task. To aid designers, the ADversary VIew Security Evaluation (ADVISE) formalism in the Möbius modeling framework was defined to provide an executable, quantifiable security model of a system under attack by an adversary. ADVISE contains an attack execution graph that includes all possible attacks against elements in the system and information on how those individual attacks link together to define attack paths to an adversary's goal states. An ADVISE model also contains an adversary profile that details an adversary's preference for or aversion to risks of cost, payoff, and detection, as well as what initial skills and assets an adversary possesses, e.g., physical access to a server room.

In the Möbius framework, the performance variables reward model allows quantifiable security metrics, e.g. time to compromise a database, to be specified on the ADVISE model. Studies in the Möbius framework offer the ability to examine multiple configurations of an ADVISE model that can vary things such as initial assets of an adversary or the difficulty of an attack step in the attack execution graph. By means of the discrete event simulator in the Möbius framework, ADVISE models are executed and approximate solutions to security metrics are calculated over many simulation iterations.

ADVISE models are manually specified by security system analysts. The models can be complex and are tedious to build for reasonably large systems. To mitigate the problem, the ADVISE meta model has been proposed. An ADVISE meta model contains a higher-level system diagram containing objects, such as components, participants, networks, and policies. connected by relationships between these objects, e.g. a UPS device may have a *poweredBy* relationship with a web server component. In addition, an ADVISE meta model contains a set of adversaries and a set of metrics that define adversary profiles and security metrics, respectively.

Given an ADVISE meta model, a set of ADVISE models can be automatically generated using details provided in the meta model in tandem with information in an ontology library that details possible components, ADVISE elements, adversaries, and metrics. The remainder of this abstract focuses on how we have laid the foundation for those four parts of the core ontology library so that the ontology can be used to model a diverse array of ADVISE meta models and also be extensible so that users can add elements to the ontology and build ADVISE meta models with those contributions. We conclude with a call for readers to participate in an alpha trial of the ADVISE meta modeling formalism.

II. Component Ontology

The system diagram of an ADVISE meta model defines instances of objects that represent components, people, policies, or any other elements of a complex system. These instances come from classes defined in the component ontology library. The library also defines a set of possible relationships that two instances can possess. For example, the fact that a Kerberos server may authenticate a web application can be encapsulated in the system diagram by the connection of a Kerberos server node with a web application node by an *authenticatedBy* relationship.

Each class in the ontology has a parent class from which it inherits. That inheritance relationship is leveraged to inherit attributes, possible relationships, and attack vectors. For example, a *server* may be vulnerable to a DDoS attack. If a *web server* class is defined as a descendant of *server*, the implication is that the DDoS attack can also be attempted on the *web server*. Further, details of that attack definition can be altered by the *web server* class; for example, perhaps a DDoS attack against a web server has a higher likelihood of success.

Because attacks are defined at each level of the component hierarchy, we believe that the generated set of attacks on instances in the system, which probably have many ancestors,

will approach a complete collection of possible attacks on a real-world component. That will greatly relieve the modeling requirements with which past ADVISE users struggled in attempting to come up with every possible attack on their system.

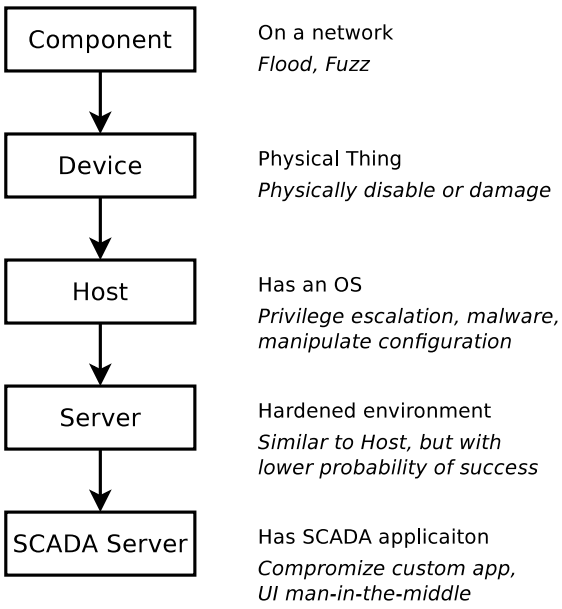

Fig. 1. A fragment of the component ontology inheritance hierarchy. The arrows indicate that the target inherits from the source. The text on the right states a quality of each element. The italicized text provides a few examples of available attacks on this component.

III. ADVISE Element Ontology

Once an ADVISE meta model has a system diagram defined, a complete attack execution graph can be generated from the system diagram, the component ontology library, and the ADVISE element library. This second library contains the possible access, skill, knowledge, and goals that are associated with each class in the component ontology. For example, the *server* component may have a *root access* access, a *root password* knowledge, or a *steal data* goal associated with it. Similarly, each class in the component library has attack step classes associated with it in the ADVISE element library, as in the *DDoS* attack mentioned earlier.

Using the attributes of instances and the relationship arcs in the system diagram, we use an ontology reasoner to find the set of all attack steps and associated access, skill, knowledge, and goals that are possible in the attack execution graph. We will take that "complete" attack execution graph, and trim it substantially using information from other parts of the ADVISE meta model.

IV. Adversary Ontology

The second part of an ADVISE model is an adversary profile. To generate that portion from an ADVISE meta model, we provide the ability to specify a set of adversaries attacking the system. Currently, in ADVISE, the modeler must fully specify an adversary's attributes, as well as initial assets, from scratch. With an ADVISE meta model, an ontology library of adversary profiles is available to choose from. The profiles provide default values that appropriately couple with the system diagram that has already been specified. For example, an insider engineer will usually have physical access to server rooms, so that access from the ADVISE element ontology will be included in the initial assets of the *insider engineer* class in the adversary ontology. However, if the system diagram does not include any server rooms, that access will be omitted from the profile if a user selects it to be included in the ADVISE meta model.

Like the component ontology, the adversary ontology library leverages an inheritance structure that inherits profile information. For example, a *nation state* may have certain interests in risks of cost, detection, and pay off. A *well-resourced nation state* may have similar interests in detection and payoff, but the class may alter its parent's preferences regarding cost because it has more resources to commit to an attack.

After the set of adversary profiles and their attributes have been selected, the "complete" attack execution graph mentioned earlier can be trimmed, starting at the attack execution graph's goals. Any goals that are not of interest to some adversary can be pruned from the graph. Further, any attack step that does not make progress to any of the remaining attack goals can be pruned off.

V. Metric Ontology

The metric ontology library defines the needed information to express quantitative security metrics for the system that is specified by the system diagram and the set of adversaries. ADVISE meta metrics are high-level, plain English definitions of system metrics with a set of attributes that define what will be measured and how. For example, a *probability computer compromised* metric would have an attribute that requires that a *computer* instance from the system diagram be selected. It could also have time bounds that allow that metric to be calculated only during a specific range of time in the simulation.

VI. Alpha Trial

Beginning in late September, 2015, our team will be performing an alpha trial of this work. We are now actively looking for participants. During the trial, we will provide access to an alpha version of Möbius with the ADVISE meta formalism as well as useful documentation for learning the tool. Hands-on support will be available to all participants. We will be in frequent communication with our alpha testers, including regularly scheduled conference calls. A mailing list and wiki will also be available for discussing progress with the alpha tool and possible improvements to either the approach or the tool.

If you are interested in participating in the alpha trial, please contact us at advise@mobius.illinois.edu.

Acknowledgment

The work described here was performed, in part, with funding from the Department of Homeland Security under contract HSHQDC-13-C-B0014, "Practical Metrics for Enterprise Security Engineering."