Multi-Aspect Security Configuration Assessment

Mirko Montanari
Department of Computer Science
University of Illinois at Urbana-Champaign, USA
mmontan2@illinois.edu

Roy H. Campbell
Department of Computer Science
University of Illinois at Urbana-Champaign, USA
rhc@illinois.edu

ABSTRACT

Evaluating the security of a computer network system is a challenging task. Configurations of large systems are complex entities in continuous evolution. The installation of new software, a change in the firewall rules, or the discovery of a software vulnerability can be exploited by a malicious user to gain unauthorized control of the integrity, availability and confidentiality of the assets of an organization.

This paper presents a framework for building security assessment tools able to perform online verification of the security of a system configuration. Heterogeneous data generated from multiple sources are integrated into a homogeneous RDF representation using domain-specific ontologies and used for assessing the security of a configuration toward known attack vectors. Different vocabularies can be defined to express configurations, policies and attacks for each aspect of the security of an organization (e.g., network security, physical security and application level security) in a modular way. By automatically extracting part of the configuration from the network system, the tool is able to detect in near real-time security threats created by configuration changes.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General

General Terms

Security, Management

1. INTRODUCTION

Computer networks are large systems with complex configurations. The settings of firewalls and hosts, the physical location of assets, and the state of applications are all part of the configuration of a system and they all interact to maintain the system secure. However, even in well-designed systems, small configuration changes might allow malicious users to control successfully or disable critical assets. This paper presents a framework for building security assessment tools able to react in real-time to changes in the system configuration. The framework partitions complex configurations into simpler modules managed by different people with different expertise and tools.

We introduce the concept of a *security aspect* as a building block for security evaluation. A security aspect addresses the security of a specific aspect of system configuration. Security aspects can be defined, for example, for network security, physical security, authorizations and application-level security. Each security aspect is composed of a domain-specific vocabulary used to describe the configuration, a set of deduction rules that describe potential attacks and their consequences and a policy that defines desired and undesired configurations. All elements of security aspects are represented using the Resource Description Framework (RDF) [16]. Ontologies define the vocabulary for the statements that describe configurations, policies and attack consequences. RDF provides a homogeneous configuration knowledge base that integrates information about all security aspects of the system.

The subdivision of knowledge in multiple aspects provides a framework for automatically extracting and integrating information generated from different sources: documents, security advisories, network management information and security assessment tools. In particular, information about frequently changing configurations of network systems can be extracted directly from the hosts using standard protocols and fed in real-time to the security analysis for validation.

The paper is organized as follows. Section 2 describes the related work in the area of security assessments. Section 3 provides an overview of our framework. Section 4 describes its evaluation. Section 5 concludes the work and provides directions for future research.

2. RELATED WORK

Evaluating the security of complex networked systems is an open research topic. In this paper we focus on the problem of evaluating the security of a network toward known attacks. The goal of the evaluation is to consider the effects that malicious users can have on the system when they take advantage of known exploits and misconfigurations. The assessment is not able to identify previously unknown ways of violating the system security. However, it is able to verify that a system is "secure enough" toward known attacks.

Attack trees [1, 7, 12, 14] have been used for representing all known sequences of actions (i.e., exploit a particular software vulnerability) that a malicious user can perform to reach a particular goal. When attacks are considered *mono-*

SafeConfig'09, November 9, 2009, Chicago, Illinois, USA.

tone, attack trees can be generated using Datalog [12]. This work uses the same reasoning algorithms described in [12] for the computation of attack trees. This work focuses on methods for the representation of knowledge about networks and on automatic extraction of configuration information for performing online security analyses.

Other work [5] proposes the use of ontologies in security assessment to formalize the knowledge about the security domain. Ontologies are used to formalize the information contained in manuals such as the NIST Security Handbook. Security ontology can be used in our framework for defining the vocabularies used to describe security aspects. However, our work introduces rules for representing complex attacks using attack trees as implicit knowledge.

Existing approaches to network configuration management such as ConfigAssure [10] and C^2 [9] introduce specific languages for representing configurations and algorithms for detecting and solving network configuration problems. However, they do not consider complex attacks (described as attack trees) in the evaluation of the correctness of the configuration. The focus of this work is on assessing the security of a system toward known attack vectors using attack trees.

3. MULTI-ASPECT EVALUATION

A system configuration is a complex entity to describe. Multiple people are involved in the management of different aspects of a large system and each of them might describe the configuration using a different set of information. An IT administrator considers a configuration as composed of the association between IP addresses and physical machines, the interconnections between network, firewall rules, and all the other data that allow the administration of the network. A control system expert sees the computer network as a set of sensors, controllers, and actuators associated with a physical plant. For a building security expert, the configuration is the locations of the physical assets that compose the system. All these interpretations are legitimate and each of them captures a partial view of the system that can be easily managed and kept up to date.

A malicious user can take advantage of interactions between different parts of the system to perpetrate attacks (e.g., physically using a machine connected to the internal network to bypass firewall protections). Hence, a complete security evaluation needs to integrate multiple interpretations of the configuration and evaluate their interactions.

We define the concept of *extended configuration* as a representation of the state of a system. The extended configuration is composed of three types of information. The first type is the explicitly defined configuration of the system. Such information can be manually defined or automatically extracted from the system. The second type is the external information used in the assessment process (e.g., vulnerability reports from NIST). The third type is the implicit knowledge deducted from the previous information. For example, the presence of a network-exploitable buffer overflow on a system component reachable from the Internet implies that, potentially, an unauthenticated user can gain access to the machine by exploiting the vulnerable element.

The building blocks of an extended configuration are *security aspects*. Each security aspect represents a partial view of the system and it is composed of three parts. First, a domain-specific vocabulary is used to represent information about a particular aspect of the system configuration. Second, a set of implication rules, defined using domain-specific concepts, describes the implicit knowledge. The rules define that the presence of certain conditions on the configuration implies other information. They can be used for representing known attack vectors, such as the exploitation of generic software vulnerability or the physical compromise of a machine, or for describing the interactions between aspects. Third, a security policy defines admissible or inadmissible conditions over the extended configuration.

Security aspects can be defined for any aspect of the configuration that has security implications. For example, an organization can define security aspects for **Network** to describe the configuration of hosts and firewalls; **Location** to describe physical locations of assets and employee access to restricted locations; **Wireless** to describe the availability of wireless networks on the organization premises; **Storage** for representing network file systems and portable storage devices; and **Role** to describe role-based access control.

3.1 Security as Knowledge Management

The description of a configuration is a representation of knowledge. Previous work [12] focused on a decidable subset of first order logic, Datalog, as its representation. However, Datalog [2] does not provide methods for formally representing the syntax of the concepts that are described and does not provide a framework for modularly representing knowledge. For this reason, we decided to use the Resource Description Framework (RDF) language to provide a formal syntax for representing security aspects. The RDF language represents information using statements (called *triples*) composed of three elements: a subject, a predicate and an object. All information about systems is expressed using triples. For example, the fact that a machine M has user U as an authorized user is specified by the triple (M `hasUser` U).

Each security aspect defines a vocabulary and a syntax for describing configurations using RDFS. RDF Schemas (RDFS) [15] express the valid syntax of RDF statements for each security aspect. RDFS provide an ontology for describing the fundamental concepts used for reasoning about security for each aspect of the security of an organization. This simplifies the process of extraction of information and provides the ability to specify security concerns using a terminology related to the domain under analysis.

3.2 Attacks as Implicit Information

The possibility of attack is represented as an implicit statement on the extended configuration. Such implicit statements can be considered as a misconfigurations that violate the organization policy and that might allow unauthorized parties to have control over the integrity, availability or confidentiality of organization assets. Implicit statements are generated by rules defined in each security aspect. As in MulVAL [12], each rule defines a generic class of attack and the result of the analysis is defined by a fixpoint semantic.

Rules are composed of a set of preconditions and a conclusion, both expressed using triple patterns. A triple pattern is a triple where subject and object are variables. A triple pattern can be matched with triples using the same predicate. A rule $r_i : P_{i,0}, \ldots, P_{i,n} \rightarrow C_i$ is composed of n triple patterns $P_{i,j}$ (preconditions) and a single triple pattern C_i (conclusion). When expressing attacks, the preconditions represent the configuration that make the attack possible, while the conclusion represents its effects.

A triple pattern can select triples potentially defined in any aspect. To support the creation of a modular framework for security evaluation, it is important to characterize rules according to the aspects over which triples pattern can match triples.

Definition 1. A rule r_i defined in aspect A_j is a ***single-aspect rule*** if all $P_{i,0} \ldots, P_{i,n}, C_i$ are defined in the same aspect A_j. Other rules are ***multi-aspect rules***

Multi-aspect rules define the relation between security aspects. If the preconditions of a rule defined in aspect A_i can match triples defined in another aspect A_j, the rule requires concepts and information described in aspect A_j for deciding if it should be fired or not. Hence, any evaluation that contains aspect A_i requires also aspect A_j to be correctly defined. We define this relation between aspects *requires*.

A particular type of multi-aspect rule can be defined as follows:

Definition 2. A multi-aspect rule r_i defined in aspect A_j is a ***projection rule*** if C_i is defined in an aspect $A_k, k \neq j$.

Projection rules create triples syntactically defined in other aspects. For example, the conclusion of a rule defined in A_i can use a predicate defined in aspect A_j. In this case, the rule is creating a new triple that can be matched by triple patterns defined in aspect A_j. Intuitively, the rule maps concepts of A_i to concepts of A_j. A relation *projects-to* formalizes this mapping. The consequence of the project-to relation is that, if an aspect A_i projects to an aspect A_j, then a security analysis that includes A_j but does not include A_i might ignore potential attacks.

Using *requires* and *projects-to*, security aspects can be related to each other in a graph. These relations support the modular composition of the security analysis.

3.3 Policy Representation

Security aspect policies define allowed and not allowed conditions over the extended configuration. Policies are expressed using the vocabulary defined in the security aspect. A typical use of a security policy is the definition of authorizations. An authorization policy specifies that an entity, such as a user, has rights of a certain kind (e.g., read, write, execute a particular function) on another object. This relation can be directly expressed using a RDF triple. The ability to define authorization policies and the use of a domain-specific vocabulary simplify the policy definition.

Definition 3. A policy is defined as a set of statements in the form:

$$\forall v_0, \ldots, v_n : P_0, \ldots, P_k \rightarrow (T_0, \ldots, T_m) \qquad (1)$$

where the set of variables $P_0, \ldots, P_k$, called precond, is a set of triple patterns defined over the variables $v_0, \ldots, v_n$. The set of variables $(T_0, \ldots, T_m)$ is called condition and it is a set of triple patterns defined over the same set of variables. Conditions can be positive or negative. A set of negative conditions is prefixed with $\neg$. All variables that appears in condition need to appear in $P_0, \ldots, P_k$.

For example, a network aspect policy can define that an unauthorized user should not control any organization asset, while a physical aspect policy can state that every authorized person needs to have access to a restricted room R. The validation of the policy can be performed by querying the extended configuration for instances of variables that match the precondition. For each found instance, another query is issued. A positive condition fails if the second query does not provide results, while a negative condition fails if the query provides results. If a condition fails, the policy is violated.

3.4 Real-Time Security Notifications

Partitioning the configuration of a system into multiple aspects simplifies the process of knowledge extraction. Information from network management protocols, security advisories and documentation are used to define the system configuration. For each information source, a new aspect, called *source aspect* is created. This aspect is composed of an ontology that mirrors the concepts originally represented in the information source. For example, XML documents can be converted into RDF triples expressed in the source aspect without changing the semantic of the documents. Concepts of the source aspect can be mapped to a security aspect using projection rules.

A real-time configuration validation is possible when data sources generate events that represent configuration changes. System logs can be used for this purpose: a process can monitor a log file for lines that represent changes in the configuration. Network management protocols such as WBEM already provide an event system through which a client can be notified of any configuration change. When a configuration-change event is received, the extended configuration is modified accordingly and policies verified again.

4. FRAMEWORK EVALUATION

The evaluation of the framework is composed of three parts. First, we apply the framework and create a model for the security evaluation of a generic organization. Second, we describe how to acquire information about the network configuration directly from machines that compose the system using WBEM protocol and the Common Information Model (CIM) [4]. Third, we show that the assessment can scale up to large networks.

4.1 Definition of Security Aspects

This section describes a model for the security evaluation of a generic organization network. For space reasons, we only provide an overview of the ontology, rules, and policies defined in the network, location, and wireless aspects.

The **network security aspect** represents the tradition network security assessment. It represents the concept of software vulnerability and encodes in rules network attacks such as locally and remotely exploitable buffer overflows. To define the network reachability graph, the network aspect includes information about connectivity between networks and firewall rules at these interconnections. A typical set of statements can state that a `ComputerSystem` A is connected to a `Network` N and provides a `ComputerNetworkService` S (bound to a `ServicePort` P) using `Software` S; S has `Vulnerability` V. Information about vulnerabilities are modeled upon the NVD data source. A rule defining an attack that uses a remote exploitable vulnerability with privilege escalation to administrator can be defined as follows:

```
(?U rdf:type core:User), (?MA net:controlledBy ?U),
(?MA net:reachService ?S), (?MB net:provideService ?S),
```

```
(?S net:software ?SW), (?SW net:hasVulnerability ?Vul),
(?Vul net:hasAccessVector net:NetworkAccessVector),
(?Vul net:hasPrivilegeEscalation net:Administrator)
 ->  (?MB net:controlledBy ?U)
```

where elements starting with '?' represent variables and prefixes `ser:`, `rdf:` before entities represent namespaces associated with security aspects. `rdf:` is defined in the RDF standard, while `core:` represents a security aspect that introduces basic security concepts. The rule specifies that: when user `U` controls a machine `MA` that can reach the service (i.e., port) `S` of machine `MB` provided by software `SW` that has a vulnerability `Vul` which allows a privilege escalation to administrator; then we can infer that `MB` can be controlled by `User` too. Other rules define the reachability graph by reasoning on firewall rules; describe the effects of vulnerabilities on the availability, integrity and confidentiality of a machine; and encode other types of exploits. A sample network-aspect policy can identify as undesired all triples describing that a `MaliciousUser` has control of a computer system of the organization through the use of: ¬((`?U rdf:type core:MaliciousUser`), (`?M rdf:type net:ComputerSystem`), (`?M core:partOf org:OrgA`), (`?M net:controlledBy ?U`)) . Using the information provided in the network aspect, the security analysis creates all known network attack trees and determines the potential capabilities of an intentional attacker that exploits system vulnerabilities.

The **location aspect** represents the concept of physical location for assets and users. Assets are associated with their locations and each user is authorized to access only specific locations. Locations are organized in an embedding hierarchy (e.g., an office room within a building). Rules implicitly define the locations to which a user has access. A user U has access to a location A if she has access to a location B that embeds A and either A is public access or she is authorized to enter A. In the extended configuration, every user U is associated with all the locations that she has access from her starting position. Any security aspect that requires knowledge about the physical locations can use the location aspect to describe it.

The **wireless security aspect** describes the interaction between mobile computers and wireless networks. The main concept is `WirelessNetwork`, a subclass of `Network`. A wireless network has a range represented using `Location`s. A predicate defines a computer system as mobile. If the system is mobile, its location is the location of the user who physically controls it. Rules define the ability of systems to connect to wireless networks. A system can connect to a network if within range and either the network is public or the user controlling the system has an appropriate access credential. The wireless aspect provides an example of an aspect that *requires* the location aspect to be defined and which produces network aspect triples (*projects-to*).

4.2 Acquisition of information from CIM

Certain system configurations, such as the ones described in the network aspect, change frequently: installation of new software, discovery of new vulnerabilities or changes in firewall policies can open new ways for attackers to compromise a system. As a demonstration of the capabilities of the framework to automatically acquire configuration and react to changes, we describe a tool that uses WBEM/CIM protocols to monitor the configuration of the network and feeds this information to the security assessment process.

We defined a new source aspect called CIM. This aspect represents the translation of CIM specifications into RDFS [6]. A client queries a WBEM server and converts the retrieved configuration in RDF. A set of projection rules relates the CIM source aspect to the network aspect. An example of one of the rules is as follows:

```
(?S rdf:typeOf cim:System),(?AC rdf:typeOf
cim:HostedAccessPoint),(?AC cim:HostedAccessPoint_System
?S), (?AC cim:HostedAccessPoint_ServiceAccessPoint ?L),
(?L rdf:typeOf cim:LANEndpoint),(?L cim:LANEndpoint_LANID
?LAN) -> (?S net:connectedTo ?LAN)
```

The rule specifies that the network concept `connectedTo` is represented in CIM as a system associated with a `SystemAccessPoint` of type `LANEndPoint`.

WBEM/CIM provides a mechanisms for event notification [3]. A client can subscribe to receive events when there are changes in the configuration, such as the addition, deletion or modification of CIM instances or classes. By subscribing to events, the security analysis system can be notified of any change in the configuration of machines. When changes occur, the extended configuration is modified and the security analysis is run again.The use of events creates a system that can quickly react to new security threats caused by configuration changes.

4.3 Performance Evaluation

The goal of this section is to provide an initial evaluation of the feasibility of reasoning based on RDF. The results demonstrate that the approach can scale to large networks even just using standard reasoning algorithms. We evaluated the framework using the XSB-like backward chaining algorithm [13] provided by the Jena 2 Semantic Web framework [8] on an Intel Xeon 2.33GHz with 3GB of RAM. The description of each system is randomly generated. Each system is composed of multiple subnetworks generated for every 100 hosts. Each host provides up to 20 services. Each software component has a vulnerability with probability 0.1. Random pairs of networks are connected using firewalls and are directly connected to the Internet with probability 0.5. Firewalls allow a random number of services. In our experiments we were able to compute the extended configuration (all attack graphs) for networks of size up to 500 nodes in less than 30 seconds, while networks up to 1000 nodes in about 2 minutes. Incremental evaluation [14] can be used to further improve such results for an event-oriented security evaluation.

5. CONCLUSIONS

This paper presents a framework for modularly representing the configuration of a system using domain-specific vocabularies with the purpose of evaluating the potential effects on the network system of the exploitation of known vulnerabilities and misconfigurations. The framework has been evaluated through the creation of a tool able to automatically extract the configuration of a network system from its hosts and react in near real-time to configuration changes. Online assessments increase the security of a network system by reducing the vulnerability time window for an attacker to exploit. Future work will address the problems of efficiently decentralizing the security evaluation on multiple machines to remove potential bottlenecks and of reducing the network load by intelligently limiting the subscriptions to the configuration-change events.

6. REFERENCES

[1] P. Ammann, D. Wijesekera, and S. Kaushik. Scalable, graph-based network vulnerability analysis. *ACM Conference on Computer and Communication Security (CCS)*, pages 217–224, 2002.

[2] S. Ceri, G. Gottlob, and L. Tanca. What you always wanted to know about datalog(and never dared to ask). *IEEE Transactions on Knowledge and Data Engineering*, 1989.

[3] Distributed Management Task Force. CIM Event Model White Paper, *Distributed Management Task Force White Paper*, 2003

[4] Distributed Management Task Force. DMTF Standards and Initiatives *http://www.dmtf.org/standards/*. Visited: 08/3/09

[5] S. Fenz and A. Ekelhart. Formalizing information security knowledge. *ACM Symposium on Information, Computer and Communication Security (ASIACCS)*, pages 183–194, 2009.

[6] D. Heimbigner. DTMF-CIM to OWL: A case study in ontology conversion. *Ontology in Action Workshop in conjunction with the International Conference on Software Engineering and Knowledge Engineering*, volume 16, 2004.

[7] S. Jajodia, S. Noel, and B. O. Berry. Topological analysis of network attack vulnerability. *Managing Cyber Threats: Issues, Approaches and Challenges*, pages 1–20, Springer, 2005.

[8] Jena Team, Jena - A Semantic Web Framework for Java, 2009. http://jena.sourceforge.net/

[9] J. Lobo, V. Pappas. C2: The Case for a Network Configuration Checking Language. *IEEE Workshop on Policies for Distributed Systems and Networks*, 29-36, IEEE, 2008

[10] S. Narain, G. Levin, S. Malik, and V. Kaul. Declarative infrastructure configuration synthesis and debugging. *Journal of Network System Management*, 16, Springer, 2008.

[11] NIST. National vulnerability database version 2.2, 2009. http://nvd.nist.gov/

[12] X. Ou. A logic-programming approach to network security analysis. *PhD Dissertation*, Princeton University, 2005.

[13] K. Sagonas, T. Swift, and D. S. Warren. XSB as an efficient deductive database engine. *ACM SIGMOD Record*, 23, 1994.

[14] D. Saha. Extending logical attack graphs for efficient vulnerability analysis. *ACM Conference on Computer and Communication Security (CCS)*, pages 63–73, 2008.

[15] W3C. RDF Vocabulary Description Language 1.0: RDF Schema, 2004 http://www.w3.org/TR/rdf-schema.

[16] W3C. Resource Description Framework (RDF), 2004 http://www.w3.org/RDF.