The STAC (Security Toolbox: Attacks & Countermeasures) ontology

Amelie Gyrard
Eurecom
Campus SOPHIATECH
450 Route des Chappes,
06410 BIOT
+ 33 (0) 4.93.00.81.92
ameliegyrard@gmail.com

Christian Bonnet
Eurecom
Campus SOPHIATECH
450 Route des Chappes,
06410 BIOT
+ 33 (0) 4.93.00.81.08
christian.bonnet@eurecom.fr

Karima Boudaoud
Laboratoire I3S-CNRS/UNSA
930 Route des Colles, BP 145
06903 Sophia Antipolis Cedex
+ 33 (0) 4.93.95.51.87
karima@polytech.unice.fr

ABSTRACT

We present a security ontology to help non-security expert software designers or developers to: (1) design secure software and, (2) to understand and be aware of main security concepts and issues. Our security ontology defines the main security concepts such as attacks, countermeasures, security properties and their relationships. Countermeasures can be cryptographic concepts (encryption algorithm, key management, digital signature, hash function), security tools or security protocols. The purpose of this ontology is to be reused in numerous domains such as security of web applications, network management or communication networks (sensor, cellular and wireless). The ontology and a user interface (to use the ontology) are available online.

Categories and Subject Descriptors

D.2.0 [**Software Engineering**]: General—*protection mechanisms*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*authentication*; K.6.m [**Management of Computing and Information Systems**]: Miscellaneous—*security*

General Terms

Security, Languages

Keywords

Security, ontology, attacks, countermeasures, semantic web, taxonomy, wireless communications, security protocols, OSI model

1. INTRODUCTION

We intent to help developers, who are not expert in security, to design secure applications and be aware of main security concepts and risks in several domains. Let's take an example where a developer has to design a secure software using heterogeneous technologies: Wi-Fi and sensor networks. Both domains have their own threats, countermeasures and protocols. Three well-known protocols have been created to protect Wi-Fi connections: WEP, WPA1 and WPA2. The developer does not know which one to use. He needs to have more information such as strengths and weaknesses of these protocols, which one is the most secured. He has the same problem with sensor networks. How to secure sensor networks? Is it possible to use Wi-Fi protocols to secure sensor networks? If we consider the RSA (Rivest Shamir and Adleman) asymmetric algorithm, used to exchange keys, it cannot be applied to sensor networks as it consumes lot of resources, whereas the LEAP (Localized Encryption and Authentication Protocol) protocol is a key management suitable for sensor networks.

To help the developer to design a secure application, we have created an ontology, called STAC (Security Toolbox: Attacks & Countermeasures) because existing ones [3, 4, 5, 1, 2] do not: (1) link similar concepts to existing ontologies, (2) indicate that attacks/countermeasures are categorized by domain and according to the OSI model, (3) describe countermeasures: their strengths, their weaknesses and if they are composed of other countermeasures, (3) specify the relationships between countermeasures and security properties (e.g., authentication) and classify them when they satisfy the same security property and (4) explain relationships between the application to secure and the countermeasures.

2. THE STAC ONTOLOGY

The STAC ontology specifies relationships between the following concepts: `Application`, `Requirement`, `Domain`, `Attack`, `Countermeasure`, `Feature`, `SecurityProperty` and the `OSIModel` (see Figure 1). We design that the `Application` to secure has `Requirements` (`SecurityProperty`, `Domain` and `DataTypeSensitive`). The `DataTypeSensitive` concept defines the type of the data to secure (`LowSensitive`, `MediumSensitive` or `HighSensitive`). We specify that a `Domain` is protected by `Countermeasures` (the `isProtectedBy` property) and cannot thwart all `Attacks` (the `hasVulnerability` property). We indicate that a domain has `Features`, and countermeasures have some strengths and weaknesses related to the features of the domain (`Advantage` and `Drawback` concepts). We define that an attack appears in an `OSIModel` layer (the `occursInLayer` property) and a countermeasure is specific to an OSI model layer (the `protectsInLayer` property). Finally, we define that countermeasures satisfy `SecurityProperties`. We classify attacks and countermeasures according to the OSI model. For example, the jamming attack occurs in the physical layer, whereas the SSL countermeasure protects the transport layer. Moreover, these attacks are classified by domain: `WebApplication`,

WWW 2013 Companion, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2038-2/13/05.

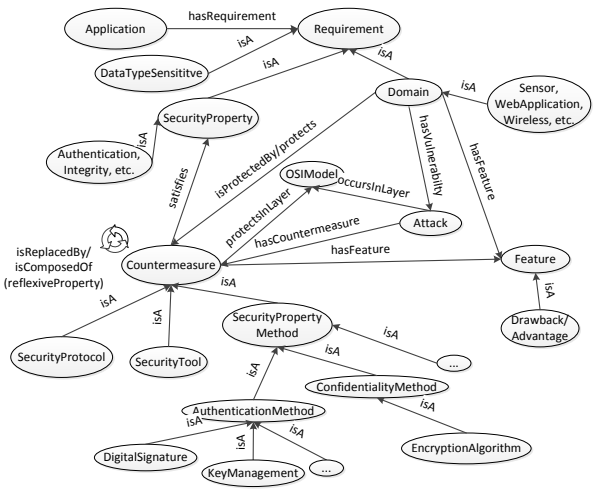

Figure 1: The top-level part of the STAC ontology

`NetworkManagement`, `Sensor`, `Cellular` (2G, 3G, 4G), `Wireless` (Wi-Fi). As for the attacks, we categorize the countermeasures by domain. For example, for the `Sensor` network domain, there are `SensorCountermeasures`, `SensorProtocols` and `SensorKeyManagements` (e.g., LEAP) and for the `WiFi` domain, there are `WiFiCountermeasures`: `WiFiProtocols` (WEP, WPA1, WPA2) and `WiFiKeyManagements`.

We define countermeasures as prevention mechanisms to thwart attacks and classify them into cryptographic concepts (`EncryptionAlgorithm`, `HashFunction`, `DigitalSignature`, `KeyManagement`), `SecurityProtocols` or `SecurityTools`. A countermeasure can be itself be composed of others countermeasures. (e.g., the VPN `isComposedOf` the IKE key management and the IPSec security protocol). To help the developer, we also define advantages (`Secured`, `LowCostDeployment`, `LowEnergyConsuming`) and drawbacks (`Deprecated`, `NotScalable`, `HighEnergyConsuming`) of countermeasures. For example, an `AsymmetricAlgorithm` (e.g., RSA) cannot be used in sensor networks because it is high energy consuming. We also indicate that the DES algorithm has been cracked. We propose the property `isReplacedBy` to replace a countermeasure by another more secured (the DES algorithm is replaced by the Triple DES algorithm). A specific `Domain` has `Features` (e.g. the sensor network domain is low energy consuming), features are related to advantages and drawbacks of countermeasures. Thanks to a reasoner, we deduce that sensor networks need a symmetric algorithm because it is low energy consuming. `SecurityProperty` is a concept that gives more information about countermeasures. We describe thirteen security properties (`Confidentiality`, `Authentication`, `Integrity`, `AccessControl`, `Privacy`, `Trust`, `NonRepudiation`, `Availability`, etc.) to indicate that countermeasures `satisfies` some of these security properties. For example, the VPN satisfies the authentication, the confidentiality and the integrity properties. We have several methods such as `LoginPassword`, `CertificateBased`, `KeyManagement`, `DigitalSignature`. To avoid to repeat that each of these methods satisfy the authentication property, we classify these methods into the `AuthenticationMethod` concept. We define as many `SecurityPropertyMethods` as `SecurityProperties`. A full description of the ontology is available online (see http://securitytoolbox.appspot.com/stac.owl)

3. IMPLEMENTATION

We have implemented the user interface[1] powered by the STAC ontology to support the developer to design a secure application. The developer navigates through the user interface to discover security concepts in a specific domain. STAC is represented in OWL and includes relationships with five other security ontologies for similar concepts or instances. The user interface proposes a menu composed of: cryptographic concepts, communication networks (sensor, wireless or cellular), security properties with their methods and a FAQ. The user interface employs the following technologies: Java, REST Web Services (Jersey), Google Application Engine (GAE), the Jena framework to manage semantic data, the SPARQL language to query data, HTML5, CSS3, JavaScript and AJAX. Through the user interface and thanks to SPARQL queries and the Jena[2] reasoner, the developers can: (1) look for all attacks and countermeasures for a specific domain (e.g., web application attacks), (2) obtain more information about countermeasures (security tools, security protocols or cryptographic concepts) such as the advantages, the drawbacks and which security properties are satisfied or (3) search attacks and countermeasures in a specific OSI model layer.

4. CONCLUSION AND FUTURE WORKS

The STAC ontology specifies relationships between the main security concepts (cryptographic concepts, security protocols, security tools) and classifies threats and countermeasures by domain and according to the OSI model. To the best of our knowledge, we are the first project proposing a semantic-based application to help the developer to design a secure application. Currently, we are working on the refinement of the user interface and the integration of a knowledge-based recommender system (constraint-based) to suggest the best solution to secure the application. Moreover, the STAC ontology will be used for our own needs to help us to secure a distributed architecture using heterogeneous communication technologies.

5. REFERENCES

[1] Security ontology. http://semanticweb.org/wiki/File:OntologySecurity.owl.

[2] Security ontology. http://preciosa.informatik.hu-berlin.de/ontology/security.owl.

[3] G. Denker, L. Kagal, and T. Finin. Security in the semantic web using owl. *Information Security Technical Report*, 10(1):51–58, 2005.

[4] A. Herzog, N. Shahmehri, and C. Duma. An ontology of information security. *International Journal of Information Security and Privacy (IJISP)*, 1(4):1–23, 2007.

[5] A. Kim, J. Luo, and M. Kang. Security ontology for annotating resources. *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, pages 1483–1499, 2005.

[1] http://securitytoolbox.appspot.com/

[2] http://jena.apache.org/index.html