CySeMoL: A tool for cyber security analysis of enterprises

Hannes HOLM, Teodor SOMMESTAD, Mathias EKSTEDT, Lars NORDSTRÖM
Royal Institute of Technology (KTH)
{hannesh, teodors, mathiase, larsn}@ics.kth.se

ABSTRACT

The Cyber Security Modelling Language (CySeMoL) is a tool for quantitative cyber security analyses of enterprise architectures. This paper describes the CySeMoL and illustrates its use through an example scenario involving cyber attacks against protection and control assets located in an electrical substation.

INTRODUCTION

Many power utilities' process control systems are built upon commodity IT solutions which exchange data with the office environment in various ways. Utilities' IT environments contain a large number of systems and system components (computers, network equipment etc. as well as bespoke and third-party off-the-shelf applications) connected to each other to make up a complex system-of-systems. To assess the security of an enterprise's system architecture as a whole an enormous amount of factors need to be considered. It is not enough to simply assess all vulnerabilities present in the computers; to assess the vulnerabilities of the larger system-of-systems one also a need to understand how these vulnerabilities relate to each other. Furthermore, to make use of this vulnerability information there is a need to know what countermeasures which are appropriate and how they should be prioritized for this specific system-of-systems.

Decision makers of enterprise security issues typically have a basic understanding of the enterprise architecture of their organization and the losses incurred if assets are successfully attacked. However, since it is a highly complicated matter, decision makers' understanding of how vulnerabilities in their architecture depend on each other and how they can be exploited often hazy. Support in terms of security theory can be obtained from security experts or literature. However, involving security experts is costly and studying literature is cumbersome. Thus, tools that help decision makers to understand how vulnerabilities relate and what countermeasures that are applicable given certain scenarios are valuable. Unfortunately, most decision makers lack reliable tool support for this type of holistic enterprise cyber security decisions.

Researchers have proposed various tools for estimating the cyber security of an enterprise network – a few significant being NETSPA [1], MulVAL [2] and the TVA-tool [3]. These tools however unfortunately have various constraints that delimit their usefulness for a power utility manager. For example, MulVAL and NetSPA are dependent on input from network vulnerability scanners (which may crash the scanned systems in unexpected ways) and TVA-tool requires the user to specify the set of attacks which the attacker is able to use (knowledge that few decision makers possess or have time to gather).

This paper describes the Cyber Security Modelling Language (CySeMoL), a modelling language and software tool which can be used for cyber security analysis of enterprises. This paper briefly describes the concepts of the tool, the reader is referred to [4] for a more detailed description.

THE CYBER SECURITY MODELLING LANGUAGE

The main objective of CySeMoL is to allow users to create models of their architectures and make calculations on the likelihood of different cyber attacks being successful. Since the model includes theory on how attributes in the object model depend on each other security expertise is not required from the user of CySeMoL. Users must only model their system architecture (e.g., services, operating systems, networks, and users) and specify their attributes (e.g., if encryption is used and if software is well patched) in order to make calculations possible.

The classes in CySeMoL includes various IT components such as *Operating System* (e.g., Windows XP) and *Firewall*, processes such as *Security Awareness Program*, and *Person*s that are users. Each entity has a set of attributes that can be either attacks steps made against the entity or countermeasures associated to it. These attributes are related in various ways. For example, the passwords of *password account* can be *social engineered* – but the likelihood of this attack being successful depends on whether the *person* owning the *password account* is in a *security awareness program*. Each attribute in CySeMoL can have the value True or False and represents either the likelihood of an attack being successful or the likelihood of a countermeasure being functional.

In total CySeMoL contains 22 entities, 102 attributes and 32 entity relationships. An overview of the modelling language can be seen in Figure 1. In Figure 1, the upper box in a class describes the countermeasures associated with it; the lower box describes the attack steps associated with it. Relationships are marked by the dashed lines between classes. For instance, a *Person* can be the owner of an *Account* (e.g., a *PasswordAccount*) and part of a *SecurityAwarenessProgram*.

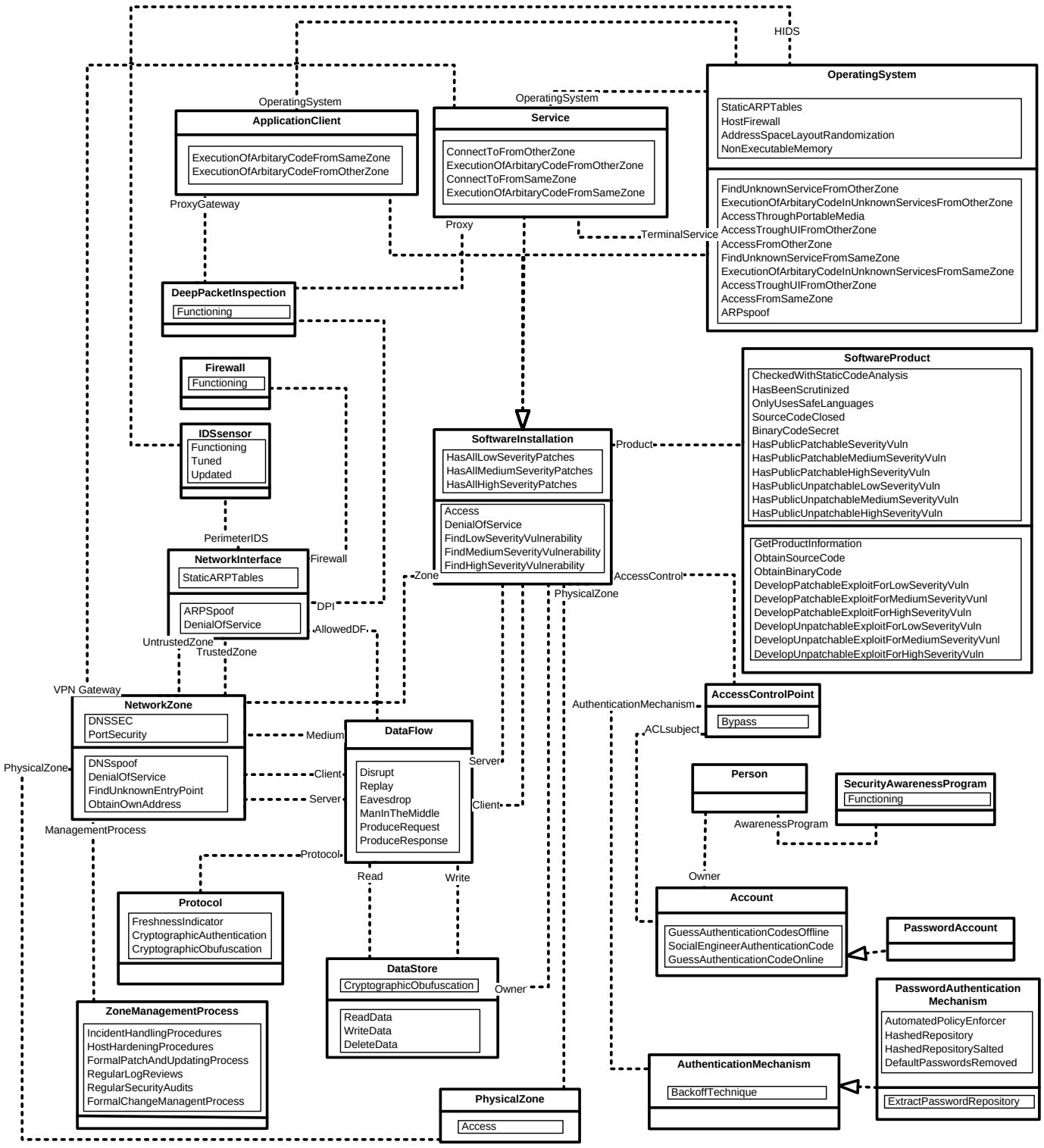

Figure 1. An overview of CySeMoL. Classes are related with dashed lines. Countermeasures are in the upper tile of classes; attack steps are in the lower tile.

The attack steps and corresponding countermeasures in CySeMoL can roughly be classified in seven different categories (cf. Table 1). The attributes related to each category were elicited based on literature and interviews with domain experts. The likelihood that an attack step is successful depends on the states of the attack steps and countermeasures that influence this attack step. Each likelihood estimate is given under the assumption that the

attacker is a professional penetration tester with one week available for the attack. For instance, the likelihood of an attacker (a professional penetration tester) discovering a new vulnerability (a.k.a. a "zero-day") in a software within one week depends on 1) whether the attacker can identify the software, whether the attacker can get its 2) binary or 3) source code, 4) if the software has been scrutinized by others, 5) if it has been completely developed in languages "safe" to buffer overflows (e.g., Java compared to C++) and 6) if it has been tested for security issues using static code analysis tools [5].

All possible combinations between influencing attributes are covered in CySeMoL. For instance, for the discovery of a new vulnerability (which has six influencing attributes) there are 2^6 different combinations available, totalling 64 possible likelihood estimates. For example, pose the following scenario: An attacker can identify the software and get its binary, but not its source code. The software has been previously scrutinized by others. It has not been developed in languages "safe" to buffer overflows. It has been tested for security issues using static code analysis tools. Given this scenario, there would be a 20% likelihood of zero-day discovery within one week [5]. However, if the source code becomes available to the attacker the likelihood will increase to 73% [5].

Most likelihood estimates (82%) in CySeMoL are deterministic, i.e., the attack is either known to be possible or known to be impossible. For instance, it is virtually impossible for an attacker to discover a zero day for a software product he or she cannot identify, get a copy of or get the blueprints for. The non-deterministic likelihood estimates are drawn from empirical studies when possible. Unfortunately, reliable empirical results are not available for all attack types CySeMoL covers. To enable quantitative analysis many non-deterministic likelihood estimates have been collected using judgment by cyber security domain experts. As the judgment of different experts can be of different quality, Cooke's classical method [6] is used to estimate the actual value of each consulted expert. In essence Cooke's method involves a knowledge test; asking each expert a set of questions for which the answer is known at the time of analysis. Judgment by an expert who is accurate and certain on these questions is more trustworthy than that of an expert who is inaccurate and uncertain.

To enable user-friendly modelling and calculation, the CySeMoL has been implemented in a software tool. This software tool is also capable of automatically generating CySeMoL models based on the results of automated network vulnerability scanners such as Nessus [7].

Table 1. Overview of categories in CySeMoL.

Category of CySeMoL	**Qualitative validation**	**Quantitative data**
Discovering new vulnerabilities	Literature and 3 experts.	Cooke's classical method applied to 17 domain experts' judgment.
Remote arbitrary code exploitation attacks	Literature, pilot study, and 3 experts.	Cooke's classical method applied to 21 domain experts' judgment.
Intrusion detection	Literature and 3 experts.	Cooke's classical method applied to 165 domain experts' judgment.
Denial-of-service attacks	Literature and 2 experts.	Cooke's classical method applied to 23 domain experts' judgment.
Exploitation of network configuration mistakes	Literature and 2 domain experts.	Literature data and judgments by four domain experts'.
Attacks on password protection	Literature and one domain expert.	A review and synthesis of password-guessing data and the capabilities of rainbow tables.
Social-engineering attacks	Literature.	Experiments on social-engineering attacks.

AN EXAMPLE USE-CASE

This section describes how CySeMoL is used in terms of modelling and analysis for an example scenario involving an substation automation architecture. For presentation purposes, this scenario is very simplified and thus not realistic.

An overview of the architecture of the substation, as modelled in CySeMoL (using the previously mentioned software), can be seen in Figure 2. The substation has a single network zone (*P&C LAN*), which has a single computer connected to it; a Human Machine Interface (*HMI*) running *Windows XP SP3* and a Remote Desktop Protocol (RDP) service for *Remote login*. The HMI has a single user (a *Technician*), who has a single password account on the system. The P&C LAN is managed by a *ZoneManagementProcess*. It is separated from the internet by a well configured *Firewall*. Any traffic to the RDP service is allowed through the firewall. Also, any traffic to and from the RDP service is encrypted using Secure Socket Layer (*SSL*). Any potential attacks are presumed to origin from the *Internet* (modelled as *Hacker computer*). Most modelled entities also have various countermeasures detailed. For instance, Windows XP SP3 is not completely developed in languages "safe" to buffer overflows (*WindowsXP_SP3.OnlyUseSafeLanguages* = False), but has been probed for vulnerabilities by various security experts (*WindowsXP_SP3.HasBeenScrutinized* = True).

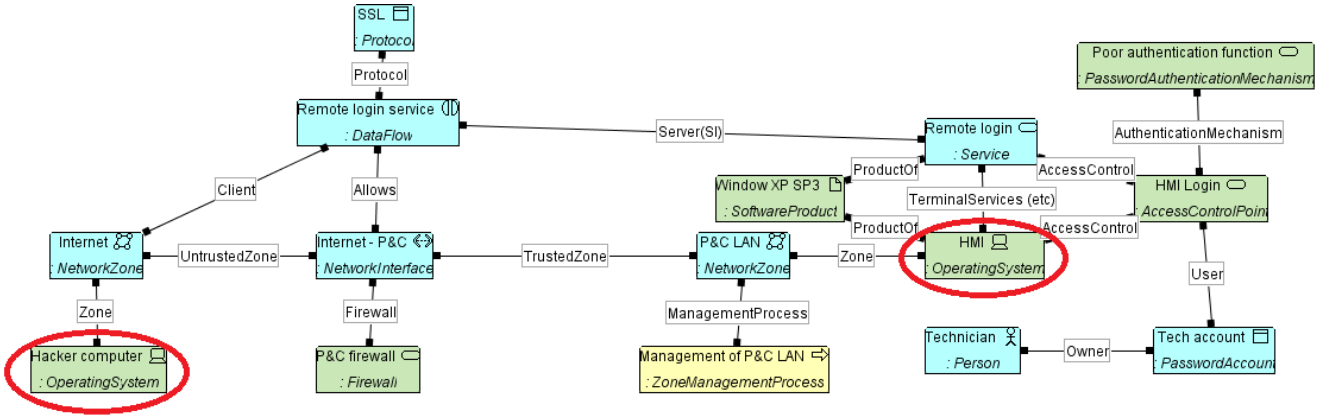

Figure 2. A CySeMoL model of the example substation automation architecture.

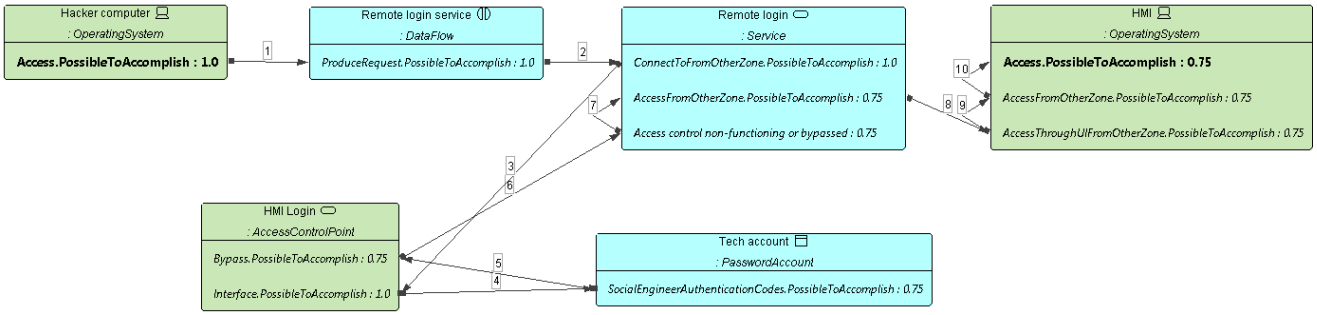

Figure 3. An attack path of a social engineering attack in the example architecture.

When the CySeMoL model has been completed the modeller specifies a source and target of the attack. In this scenario, the source is the Hacker computer and the target is to gain access to the HMI. When these constraints have been specified the modeller presses can automatically have all possible attacks possible for the scenario calculated by the tool. For the present scenario, 30 different attack paths (chains of attacks) are available. The most likely attack path has a 75% likelihood of success (cf. Figure 3). The essence of this attack path involves social engineering the technician into providing his or her credentials for the HMI and then use the remote login service to access it. Now pose that the enterprise wish to evaluate the likelihood of this attack path given that the technician undergoes security awareness training. To enable this analysis, the modeller adds the entity *SecurityAwarenessProgram*, sets its state to True, relates it to *Technician* and runs the analysis again. The likelihood of the social engineering attack being successful (step 4 in Figure 3) is now reduced to 30%.

CONCLUSIONS AND FUTURE WORK

The CySeMoL enables cyber security analyses of enterprise architectures without requiring any major cyber security knowledge of the modeller. However, the model is still very much a prototype. Currently, we are working on extending its functionality to include, e.g., time estimates for different attacks, web application attacks [8] and network vulnerability scanning [9]. We are also planning to conduct more case studies to estimate the usability and ease of use of CySeMoL.

REFERENCES

[1] R. Lippmann, “Netspa: A network security planning architecture,” Massachusetts Institute of Technology, 2002.

[2] J. Homer, K. Manhattan, X. Ou, and D. Schmidt, “A Sound and Practical Approach to Quantifying Security Risk in Enterprise Networks,” Kansas, 2010.

[3] S. Noel, M. Elder, S. Jajodia, P. Kalapa, S. O’Hare, and K. Prole, *Advances in Topological Vulnerability Analysis*. Washington, DC: IEEE, 2009, pp. 124–129.

[4] T. Sommestad, M. Ekstedt, and H. Holm, “The Cyber Security Modeling Language: A Tool for Assessing the Vulnerability of Enterprise System Architectures,” *IEEE Systems Journal*.

[5] T. Sommestad, H. Holm, and M. Ekstedt, “Effort estimates for vulnerability discovery projects,” in *HICSS’12: Proceedings of the 45th Hawaii International Conference on System Sciences*, 2012.

[6] R. Cooke, *Experts in Uncertainty - Opinions and Subjective Probability in Science*. New York, New York, USA: Open University Press, 1991.

[7] M. Buschle, H. Holm, T. Sommestad, M. Ekstedt, and K. Shahzad, “A Tool for automatic Enterprise Architecture modeling,” in *Proceedings of the CAiSE Forum 2011*, 2011, pp. 25–32.

[8] H. Holm, M. Ekstedt, and T. Sommestad, “Effort estimates on web application vulnerability discovery,” in *Hawaii International Conference on System Sciences 46*, 2013.

[9] H. Holm, T. Sommestad, J. Almroth, and M. Persson, “A quantitative evaluation of vulnerability scanning,” *Information Management & Computer Security*, vol. 19, no. 4, p. 2.