www.ijemr.net ISSN (ONLINE): 2250-0758, ISSN (PRINT): 2394-6962

Volume-7, Issue-3, May-June 2017

International Journal of Engineering and Management Research

Page Number: 146-154

Security Estimation Framework for Development of Secure Software and Cyber Attacks

Syed Mohd. Shahe Alam[1], Prof.(Dr.) Sanjay Kumar Singh[2], Dr. Suhel Ahmad Khan[3]
[1]Research Scholar, AIIT, Amity University, Lucknow, Uttar Pradesh, INDIA
[2]Head of Department, AIIT, Amity University, Lucknow, Uttar Pradesh, INDIA
[3]Assistant Professor, Department of Computer Applications, Integral University, Lucknow, Uttar Pradesh, INDIA

ABSTRACT

To design, build and deploy secure systems, we must integrate security into our application development life cycle and adapt current software engineering practices and methodologies to include specific security-related activities. Developers enforces security measures during design phase of software development processes which may end up in specifying security related architecture constraints that are not really necessary. To eliminate this problem, we propose a Framework for Security Engineering Process that involves converting security requirements and threats into design decisions to mitigate the identified security threats. The identified design attributes are prioritized and a security design template is prepared. We have stored various cryptographic techniques and their analytic attributes in the repository to find out the specific cryptographic technique that would eventually help in the later stages of the design process by eliminating unnecessary design constraints in a particular scenario.

Keywords-- Security engineering process Security requirement engineering Security design engineering Security design template Cryptographic techniques Cryptographic attacks

I. INTRODUCTION

The problem of modeling and analyzing software architectures for critical systems is usually addressed through the introduction of sophisticated modeling notations and powerful tools to solve such models and provide feedback to software engineers. However, non-functional attributes are often analyzed in isolation. For example, performance models do not usually take into account the safety of a system, as well as availability models do not consider security aspects, and so on. An early but relevant exception in this domain has been the definition of perform ability that combines performance and availability aspects into the same class of models. With the increasing variety and complexity of computing platforms, we believe that the task of jointly analyzing non-functional attributes to study possible dependencies is becoming a critical task for the successful development of software architectures.

Security engineering focuses on processes and methods for implementing security in software and related systems. The rapid development of internet based software systems which maintain sensitive information and carry out critical activities has initiated an increasing attention on how to build secure software. Current software development processes enforce security measures during design phase which may end up in specifying security related architecture constraints that are not really necessary. As a result the final software system may use inefficient mechanism with increased cost.

This leads to development of security measure at the early phase of SDLC (Software Development Life Cycle). Fires mith [2] defined security requirement as high level requirements that gives specification of system behavior and distinguish these from security related architectural constraints so that requirement engineers can discover true security requirements. There are proposal on discovering and eliciting of security requirements like abuse case, misuse case, common criteria, attack trees. The root cause of security requirements is the mitigation of different types of risk associated with the threats. To prioritize the elicited security requirements, one needs to measure underlying risks. There are also risk management methods like EBIOS, OCTAVE, CORAS, and CRAMM which enforce security levels based on risk measure. But these proposals of security engineering are not integrated in conventional software development process.

With the large scale expansion of network connectivity, there has been a rapid increase in the number of cyber-attacks on corporations and government offices

resulting in disruptions to business operations, damaging the reputation as well as financial stability of these corporations. The recent cyber-attack incident at Target Corp illustrates how these security breaches can seriously affect profits and shareholder value. According to a report by Secunia, the number of reported security vulnerabilities in 2013 increased by 32% compared to 2012. However in spite of these increasing rate of attacks on corporate and government systems, corporations have fallen behind on ramping up their defenses due to limited budgets as well as weak security practices. One of the main challenges currently faced in the field of security measurement is to develop a mechanism to aggregate the security of all the systems in a network in order assess the overall security of the network. For example INFOSEC has identified security metrics as being one of the top 8 security research priorities. Similarly Cyber Security IT Advisory Committee has also identified this area to be among the top 10 security research priorities

II. PREVIOUS WORK

Selection of inappropriate software package and security modules may be costly and also adversely affect business processes and functioning of the organization. The common security measures (security goals) are to secure the assets of the organization. Assets are generally defined as anything that has value to the organisation and needs protection from intruder or accidental detriment by recognised actors of the system. Traditionally it is known as information security and is expressed as confidentiality, integrity and availability of information in an organisation which is called as CIA triad.

2.1 Security engineering

Security engineering is a complex process consisting of different security-related activities. Security-related activities[13] include identifying security requirements, prioritizing and management of security requirements, security design, implementing security mechanisms, security testing. With true security requirements specified, most appropriate design decisions can be taken. The design phase specifies how the identified requirements (gathered from the security requirement engineering phase) can be implemented in a given environment and it also proposes an overall structure of the software from a security perspective. Getting a systematic and well organized structure of security functionality and their design decisions is the main goal of this phase. BLP (Bell-La Padula model) was a famous security model to achieve multilevel security goal developed in 1973. The aim of this model was to set rules controlling the information flow to protect sensitive data from unwanted disclosure. BLP model enforced mandatory access control policy based on sensitivity label that was a combination of hierarchical classification levels and non-hierarchical categories. It was adopted by US Department of Defence as the foundation of Trusted Computer System Evaluation Criteria and then extensively applied in information security areas, especially in the designation of secure operating systems [24]. In 2002, an aspect oriented design technique is proposed to model and integrate security concerns into design by weaving the aspects in a primary model. A design aspect can be modelled from a variety of perspective. In this paper only static and interaction aspects views are considered. This paper describes steps for weaving an aspect in primary model. But the technique fails to address the impact of the security concern on each design unit with respect to a given application environment. Also it does not focus on any well defined framework through which developers can make design decisions to develop efficient cost effective secure system. In 2005, Apvrilla et.al proposes a design methodology using security patterns and design model PICO using UML sec. This paper describes high-level principles for building more secure softwares, such as secure by design, secure by default, secure in deployment and discusses experience with its implementation across Microsoft software. But the software should be architected, designed and implemented so as to protect itself and the information it processes and resist attacks in a particular application environment. The emergence and fast development of trusted computing technologies in recent years give us some new clues to multilevel security issues and design principles, but no proper framework is found to design secure software.

2.2 Cryptographic techniques

To design a secure system, it is essential to identify the kinds of threats and the mechanisms that can be used to protect the system against such threats. Cryptography is a systematic way to secure data from adversaries. A sensitive object (asset) requires encryption, authentication and access control to protect against unauthorized modification. Symmetric or secret key cryptography has been in use for thousands of years and includes any form where the same key is used both to encrypt and to decrypt the text involved. In asymmetric or public key cryptography, this is not an issue in the same way. Two keys (private and public), mathematically related, are used and work together in such a way that plain text encrypted with the one key can only be decrypted with the other. As private key is not shared by the individual, the system avoids the risk of compromising security. Most advanced techniques are based on ECC and HECC. For authentication purpose sometimes hash functions are also used. Hash functions are also called one-way function or collision-resistant one-way function. A hash function takes a message of any size and computes a smaller, fixed-size message called a digest or hash.

2.3 Cyber security methods

A number of ideas can be found on how cyber security should be assessed. Some ideas concern how security measurements should be defined and operation ali zed. Examples include the ISO/IEC standard 27000-4 [18]

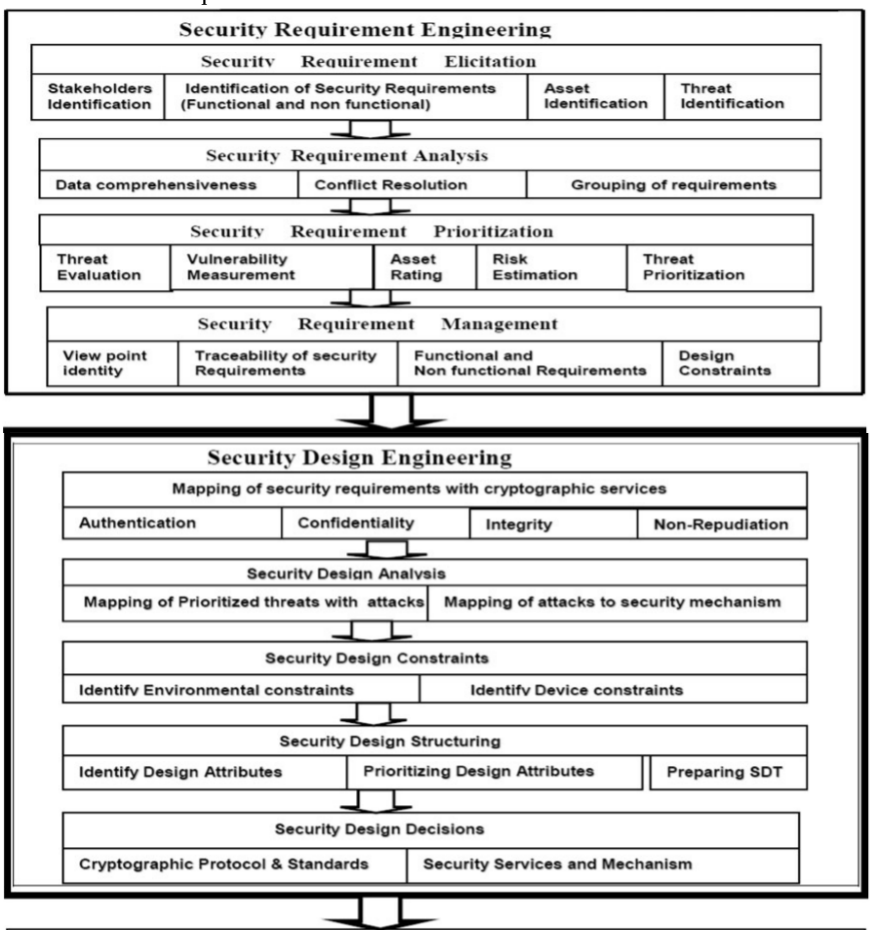
and NIST's security metric guide. These publications describe how an organization should develop and maintain a measurement program, but do not define the actual measurements that should be made or what different measurement values mean in terms of security. In addition to these there are general qualitative models that describe variables (or concepts) in the security domain and how these concepts relate to each other. CORAS contains a meta model over the security field to support assessments made using the CORAS method, Common Criteria has a conceptual model over variables (or concepts) a security assessment needs to consider and several similar qualitative models are available. For instance, are generic alternatives and are alternatives with a particular focus on SCADA systems that control energy systems. These methods, security meta models, conceptual models and technical reference models can support cyber security assessments and be used to define operational cyber security metrics. However, they require a substantial mental effort from their user – the user must identify what to measure and how important this is for the IT system's cyber security.

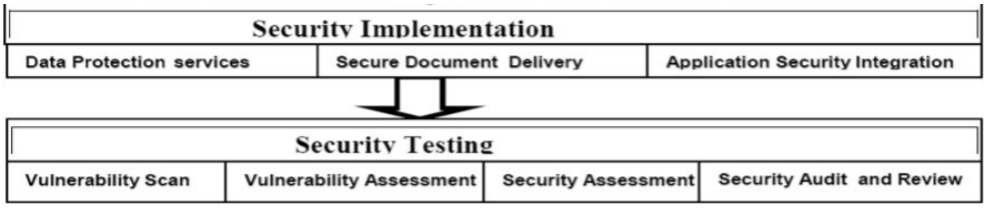
2.4. Cyber Situation Awareness

Tim Bass first introduced the concept of cyberspace situation awareness and built a framework for it which laid the foundation for subsequent research in Network Security Situational Awareness [33]. In the framework was extended to model security at a large scale level where existing techniques have been integrated to gain richer insights. Researchers have also proposed many evaluation models and algorithms for NSSA to reflect the security environment and capture the trends of changes in network state. The drawback with most of these NSSA models is that they don't adopt a consistent integrated framework for describing the relationships between the vulnerabilities in the network nor do they use an open scoring framework such as CVSS for analyzing the dynamic attributes of a vulnerability using stochastic modeling techniques

III. PROPOSED FRAMEWORK FOR SECURITY ENGINEERING

Security engineering deals with security-related activities which include identifying security requirements, prioritizing and management of security requirements, security design, implementing security mechanisms, security testing. The proposed framework for overall security engineering process (SEP) is shown in Figure below:





Firgure 1: Framework for security engineering process

The concept of this framework for security engineering process is an attempt to propose a design framework taking the view of stakeholders as well as environmental constraints in the earlier software development phases [21]. We now discuss in detail each activity of this proposed framework.

3.1 Security requirement engineering

This phase involves discovering security requirements, eliciting, analyzing and managing them. It consists of four different stages: Security Requirement Elicitation, Security Requirement Analysis, Security Requirement Prioritization and Security Requirement Management. All the different activities perform in each stage of this process are explained below:

3.1.1 Step 1: security requirement elicitation

In Security Requirement Elicitation phase different tasks are performed as explain below:

(i) Identify various stakeholders of the system using view-point analysis. We have identified the various abstract classes of actors as direct and indirect actors. Direct actors are those who directly interact with the system such as human, software system and hardware devices. Indirect actors refer to developers who develop software and people who regulate application domain.

(ii) Identify the functionalities of each actor conceptualized in the previous step and also determine associated non-functional requirements.

(iii) Identify the threats associated with each of the functional requirements or data which is used by the functionality.

(iv) Define the security requirements such as authentication, integrity, non-repudiation etc. to mitigate these threats.

3.1.2 Step 2: security requirement analysis

The various tasks perform in analyzing the security requirements are as follows:-

(i) Checking for completeness We make a check list to check that the elicited security requirements have mitigated all the threats to the functionality of the system.

(ii) Conflict resolution We resolve the contradictions that may exist in the security requirements elicited from different viewpoints.

(iii) Grouping of requirements This step consists of identifying the security requirements that can be grouped together. 3.1.3 Step 3: security requirement prioritization As security requirements are to mitigate threats and avoid vulnerability and risk, they will be prioritized on the measure of threat, vulnerability and risk. So prioritization is done in following two steps: Evaluation of threats Prioritization of security requirement.

In the first step, we will evaluate the threats based on the estimated risk value. For this we have to perform the following tasks:

(i) Threat assembling After identifying the threats, a repository of the threats will be developed as in common criteria based approach. Actor profiles will be maintained also in this repository. Thus predefined threats can be retrieved from the repository according to the profile of the actor.

(ii) Threat rating After threat assembling, we have assigned a value of each threat according to CRAMM.

(iii) Vulnerability measurement Assigning value to corresponding vulnerability.

(iv) Asset rating Identify the concerned affected asset and give them a value.

(v) Estimate the value of risk We can measure risk as Risk = value based on measure of (Threat, Vulnerability, Asset). After threat rating, assigning vulnerability value and asset value, we will use the table given by the CRAMM.

In the second step, we will prioritize the security requirement after identifying threats. Initially we have identified the measures of risk to all the threats and prioritize them based on value of risk. After finding out the high prioritized assets that are involved with the particular security requirements, we calculate the priority of security requirement just from the value of threat priority.

3.1.4 Step 4: security requirement management

As security requirement also evolve along with functional and nonfunctional requirements, it is necessary to maintain the information about traces of each security requirements and its associated attributes in this phase. The techniques for requirement management presented in can be used for this activity. There are different types of traceability information that must be maintained for the management of security requirements.

3.2 Security design engineering

This phase deals with designing a software structure that realizes the specification. So depending upon the identified security requirements, we identify the cryptographic services to mitigate the identified security threat of the system. Bad decisions made during the design phase can lead to design flaws that can leave the system vulnerable to security threats, so we focus on the design phase through a set of systematic design activities mainly

identification of cryptographic services, design structuring and finally design decisions. The steps of this process are explained below.

3.2.1 Step 1: mapping of security requirements with security services

After the security requirements have been identified, we proceed to the design phase of the security engineering process i.e. prioritized security requirements are mapped with security services like confidentiality, integrity, authentication and non-repudiation services. The different types of security requirements proposed by Fires mith are mapped to the different security services provided by cryptography. This would eventually help in the later stages of the design process, by specifying which cryptographic techniques would be suitable in a particular scenario. After the security services have been identified for the particular security requirement, we proceed to the next activity i.e. security design analysis.

3.2.2 Step 2: security design analysis

This step will define what the prioritized threats are and which assets are affected by these threats. This step consists of two sub steps as explained below: (2a) Mapping of the Prioritized threats with related attacks. In security requirement prioritizing process we identify different threats and prioritized the security requirements according to threat analysis. Now in this step, we first identify the threats which affect the assets with high value. Then we identify what type of attacks can be caused by these threats. Before developing this framework, we have done a literature survey for all security mechanisms. From this knowledge based repository we will map the security requirements with their related attacks. Suppose for authentication requirements most affected assets are smart card information, login information, account information. Now the mapping Table 4 shows how the identified threat and attacks (highlighted in italics) are related with security requirements. (2b) Mapping of attacks to security mechanism (cryptographic techniques). In this step we map the security attacks with the available techniques of cryptography and calculate the impact of these attacks. From our literature survey, the security analysis result is shown in Table 5, for a password based authentication in wireless network.

3.2.3 Step 3: identifying security design constraints

A very common cause of protocol failure is that the environment changes, so that assumptions that were originally true no longer hold and the security protocols cannot cope with new environment. A security environment describes the context in which the software is expected to evolve. The environment affects the kind of threats the application is likely to encounter. This is only because each environment has some design constraints. So before design structuring, first we have to find design constraints. The Environmental Constraints of the target deployment system is considered here depending on whether the system would be implemented on a wireless/mobile/mobile ad hoc environment. For a web based system in wireless network, there are many communicational constraints (like channel capacity, bandwidth, power, through put etc.) and computational constraints (like memory, encryption speed, energy etc.). Suppose some important information is transmitted from a mobile phone in a mobile network. One of the design constraint (encryption speed) here plays an important role for selection of cryptographic technique. While comparing the encryption speed of symmetric/asymmetric ciphers (shown in Tables 1, 2) on that device, some of the suitable crypto techniques are DES, 3DES, CAST, RSA etc. Now the design attributes will help us to select the best suitable technique.

3.2.4 Step 4: security design structuring

In this activity, different design attributes are identified which affects the selection of cryptographic protocols. The sub steps are explained below: (4a) Identifying design attributes and prioritizing them. While identifying them we have to first look whether the system would be implemented on a wireless/mobile/mobile ad hoc or any other environment. The design attributes like cost, implementation platform etc. greatly affect our design choices because only a subset of cryptographic algorithms can work efficiently on constrained environments. Further cryptographic algorithm would differ depending upon the service requirement. For example, symmetric key algorithms like AES, 3DES would be more suitable for confidentiality service requirement as they are 1000 times faster than asymmetric key algorithms like RSA which are less efficient for large plain text encryption. But asymmetric key algorithm like ECC is more suitable in constrained devices with limited memory/processing power/energy etc. for its short key size. Mainly we separate the design attributes (listed in Table 6) on the basis of the devices used because their priorities are different for High-end and Low-end devices. (4b) Preparation of security design template (SDT). After the security requirement and threats have been identified in the requirement phase and security services and design attributes identified in the first phase of the design process, we proceed with the next step in which a security design template (SDT) is prepared to take care of each security requirement. A design template is shown in next section. This template will store each specification of the design constraints and design attributes of a particular environment for further processing.

3.2.5 Step 5: security design decision

This is the final step through which we can come to the design decisions. We propose knowledge based approach to judge the best suitable security protocol depending upon the design attributes supplied in the Security Design Template. We have stored various cryptographic techniques and their analytic attributes in the repository. After impact analysis a decision is made for choosing the optimum security protocol for system under

consideration. A detailed survey has been made for collecting data of different security protocols available in literature. These templates are used to access different data for selection process in a systematic manner. In the next section, we explain how these steps will be applicable in a particular environment (smart card based banking system).

3.3 Security implementation

In this phase all functionalities are implemented incorporating design decisions of the system. This includes implementing specific techniques that are suggested in the design phase of the security engineering process. We have elaborated it with an example in subsequent Sects. 4 and 5.

3.4 Security testing

It involves evaluating the system security and determining the adequacy of security mechanisms, assurances and other properties to enforce system security policies. In this phase different tasks like vulnerability scan, vulnerability assessment, security assessment, security audit and review etc. are performed which can be done in future work.

IV. CYBER-SECURITY ANALYTICS FRAMEWORK

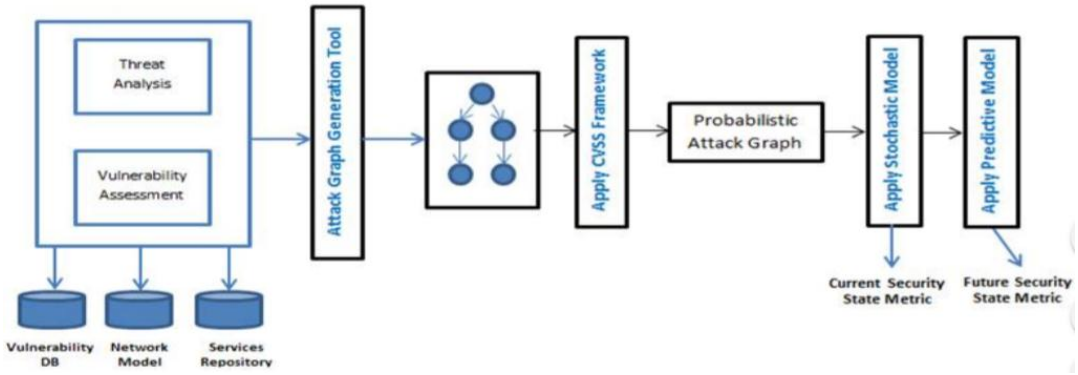


Figure 2: Cyber Security Analytical Framework

We also explore the concept of modeling the Attack graph as a stochastic process. In, we established the cyber-security analytics framework (Figure 3) where we have captured all the processes involved in building our security metric framework. By providing a single platform and using an open vulnerability scoring framework such as CVSS it is possible visualize the current as well as future security state of the network and optimize the necessary steps to harden the enterprise network from external threats. In this paper we will extend the model by taking into account the temporal aspects associated with the individual vulnerabilities. By capturing their interrelationship using Attack Graphs, we can predict how the total security of the network changes over time. The fundamental assumption we make in our model is that the time-parameter plays an important role in capturing the progression of the attack process. Markov model is one such modeling technique that has been widely used in a variety of areas such as system performance analysis and dependability analysis. While formulating the stochastic model, we need to take into account the behavior of the attacker. In this paper, we assume that the attacker will choose the vulnerability that maximizes his or her probability of succeeding in compromising the security goal.

4.1. Architecture

Figure 3 shows a high level view of our proposed cyber security analytics architecture which comprises of 4 layers where each layer builds upon the previous one below.

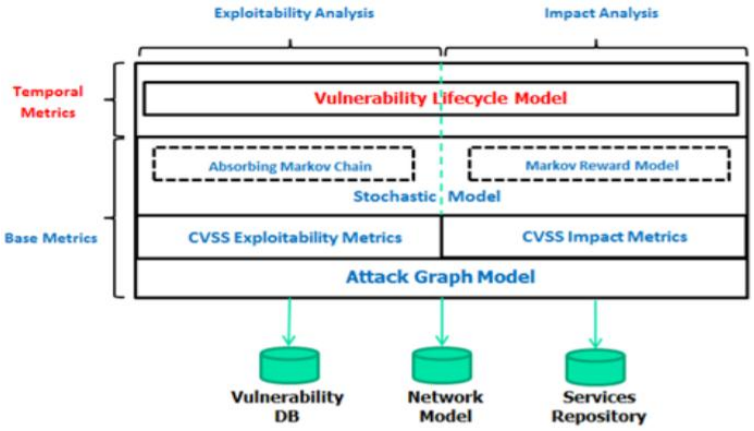


Figure 3: Cyber Security Analytical Architecture

Layer 1 (Attack Graph Model):

The core component of our architecture is the Attack Graph Model which is generated using a network model builder by taking as input network topology, services running on each host and a set of attack rules based on the vulnerabilities associated with the different services.

Layer 2 (CVSS Framework):

The underlying metric domain is provided by the trusted CVSS framework which quantifies the security attributes of individual vulnerabilities associated with the attack graph. We divide our security analysis by leveraging two CVSS metric domains. One captures the exploitability characteristics of the network and the other analyzes the impact a successful attack can have on a corporations key assets. We believe that both these types of analysis are necessary for a security practitioner to gain a better understanding of the overall security of the network.

Layer 3 (Stochastic Model):

In this layer relevant stochastic processes are applied over the Attack Graph to describe the attacks by taking into account the relationships between the different vulnerabilities in a system. For example, in our approach, we utilize an Absorbing Markov chain for performing exploitability analysis and a Markov Reward Model for Impact analysis.

Layer 4 (Vulnerability Lifecycle Model):

In order to account for the dynamic/temporal security properties of the vulnerability, we apply a Vulnerability Lifecycle model on the stochastic process to identify trends and understand how the security state of the network will evolve with time. Security teams can thus analyze how availability of exploits and patches can affect the overall network security based on how the vulnerabilities are interconnected and leveraged to compromise the system. We believe that such a framework also facilitates communication between security engineers and business stakeholders and aids in building an effective cyber-security analytics strategy.

V. IMPLEMENTATION AND EVALUATION OF THE FRAMEWORK

The framework which we described in the previous section is used to construct a tool which we call the Security Design Engine (SDE). This tool can be used to judge the best suitable security technique depending upon the design attributes supplied in the 'Design Template'. To develop this tool we use Microsoft.NET framework that runs primarily on Microsoft Windows. It includes a large library and provides language interoperability (each language can use code written in other languages) across several programming languages.

6 Performance evaluation

Now we will evaluate our framework with the other models available in the literature such as AOD [12], PICO in different phases of SDLC. Our evaluation parameters are different phases of SDLC such as security requirements analysis, security design and implementation. In security requirement engineering phase, we have (i) elicited security requirements using viewpoint oriented approach, (ii) specified different type of security requirements such as authentication requirements, privacy requirements etc., (iii) prioritized these security requirements using techniques of CRAMM. Prioritization helps us to focus a vital security issue in case of conflicting requirements and implement security in optimal manner. Whereas other models like PICO and AOD have elicited security requirement as threat model or attack tree, they have not specified and prioritized them.

In security design engineering phase, we convert security requirements and threats (which are identified during the security requirement elicitation stage) into design decisions to mitigate the identified security threats. We adopt a structural approach for design engineering. Our decisions are based on analysis of environment constraints, design attributes and possible attacks on threats. Our approach generates a template based on this analysis to enable security engineer to choose optimal cryptographic services. This chosen security mechanism is just appropriate to mitigate possible attack based on environment. Hence it not only mitigates the threats/attacks, but also does not unnecessarily constraint the availability of the system. But other models like PICO and AOD etc. take ad hoc design decisions without considering design attributes and other factors.

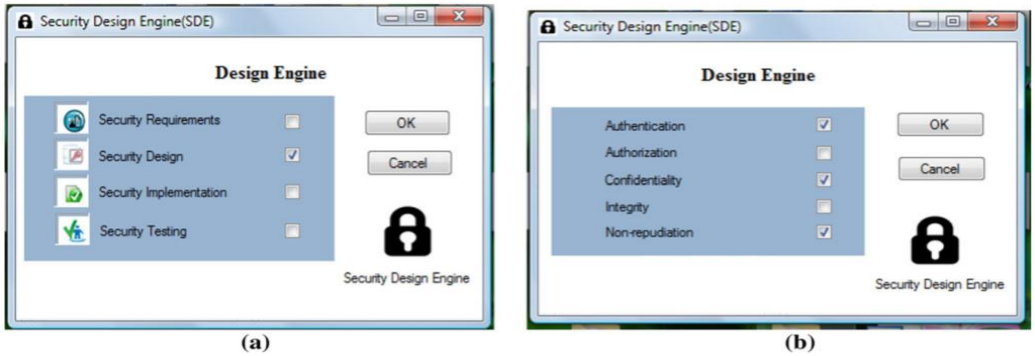
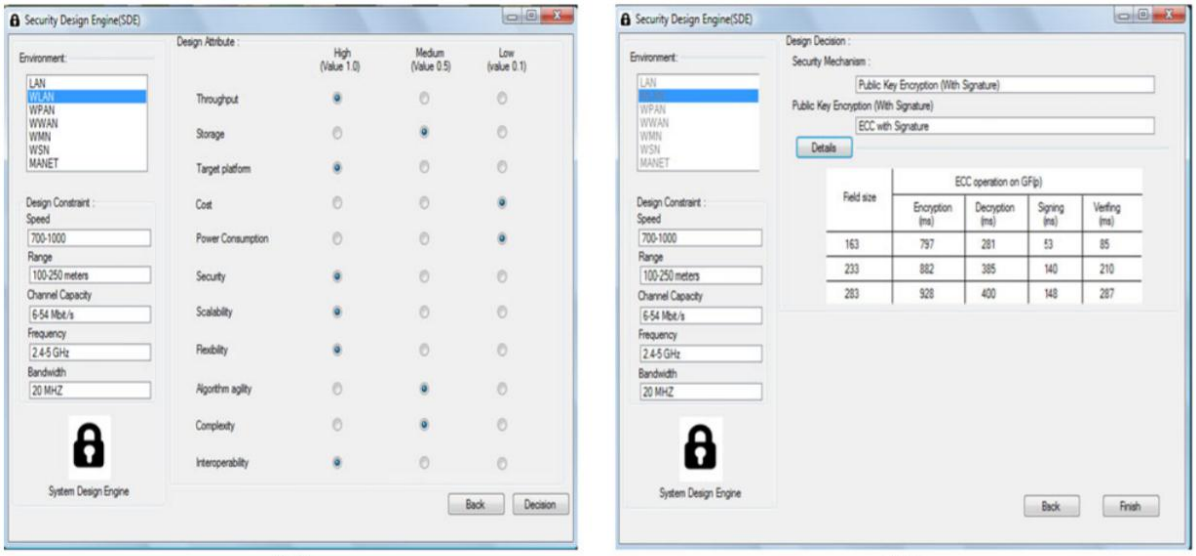


Figure 4(a) Security Engine Design, 4(b) Selection of Security Requirements for a particular Environment

ISSN (ONLINE): 2250-0758, ISSN (PRINT): 2394-6962



(a) (b)

Figure 5(a) Selection of Design Attributes for a Particular Environment 5(b) Design Decision for Above Specified Environment

The advantage of these design decisions are reflected in our implementation mechanism. While implementing design decisions, we have considered parameters like key size, encryption/decryption time. In this phase also other models have taken no specific approach considering the above parameters. In our framework, we focus on design decisions and for this we have done all the steps like mapping of prioritized threats with attacks, identifying design constraints in a particular environment which other methods have not yet considered. While selecting a key size for secure data communication, The above comparisons clearly state that our framework is a better approach to find a suitable design decision to develop efficient cost effective secure system.

VI. CONCLUSION

We have developed a Framework for Security Engineering Process and Cyber Security with a strong focus on security design engineering. We insert security concerns in each step of the life cycle. In our framework the different types of security requirements are mapped to the different security services. The identified design attributes are prioritized and a security design template is prepared to find out the specific cryptographic techniques in a particular scenario. This framework is helpful to the developers to identify and implement the appropriate cryptographic technique efficiently in a particular environment. This process is coherent with the conventional software engineering process so that eliciting security requirements and security design become an integral part of system engineering and security engineering. We also used here a realistic network to analyze the merits of our model to capture security properties and optimize the application of patches.

REFERENCES

[1] Fabian B, Gurses S, Heisel M, Santen T, Schmidt H (2010) A comparison of security requirement engineering methods. RequirEng 15:7–40
[2] FiresmithDonaldG (2003) Engineering security requirements. J Object Technol 2(1):53–68
[3] Dermott JM and Fox C (1999) Using abuse case models for security requirements analysis. Department of Computer Science, James Madison University, pp 55–64
[4] Alexander IF (2003) Misuse cases, use cases with hostile intent''. IEEE Software, pp. 58–66
[5] Guttorm S, Opdahl AL (2005) Eliciting security requirements with misuse cases. RequirEng 10:34–44
[6] Ware M, Bowles J, Eastman C (2006) Using the common criteria to elicit security requirements with use cases. IEEE Computer Society
[7] Ellison RJ (2005) Attack trees. Software Engineering Institute, Carnegie Mellon University
[8] EBIOS (2004) Expression of need and identification of security objectives. DCSSI, France
[9] Alberts C, Dorofee A (2001) OCTAVE Method Implementation Guide v2.0. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. http://www.cert.org/octave
[10] CORAS: A Platform for risk analysis of security critical systems. IST-2000-25031. http://www.nr.no/coras/
[11] The logic behind CRAMM's assessment of measures of risk and determination of appropriate countermeasures, www.cramm.com. Accessed 23 April 2007

[12] Georg G, Ray I, France R (2002) Using aspects to design a secure system, ICECCS. IEEE Computer Society, Washington, pp. 117–126
[13] Apvrille A, Pourzadi M (2005) Secure software development by example. IEEE SecurPriv 3(4):10–17
[14] Lipner S, Howard M, (2005) The trustworthy computing security development lifecycle'' security engineering and communications, security business and technology unit, Microsoft Corporation
[15] Giorgini P, Manson G, Mouratidis H, Philip I (2002) A natural extension of tropos methodology for modelling security. Workshop on Agent-oriented methodologies, OOPSLA
[16] Van LamsweerdeA (2004) Elaborating security requirements by construction of intentional anti-models, ICSE'04. IEEE Computer Society, Washington, pp 148–157
[17] Mayer N, Heymans P, Matulevic˘ius R (2007) Design of a modelling language for information system security risk management, RCIS, pp 1–11
[18] Haley CB, Laney R, Moffett JD (2008) Security requirements engineering: a framework for representation and analysis. IEEE Trans Software Eng 34(1):133–153
[19] Gupta D, Agarwal A (2008) Security requirement elicitation using view points for online system. International conference on emerging trends in engineering and technology, IEEE Computer Society, pp 1238–1243
[20] Gupta D, Jaiswal S (2009) Security requirement prioritization, Software Engineering Research and Practice, pp 673–679
[21] Chatterjee K, Gupta D, De A (2011) A framework for security design engineering process. In proceedings of fifth international conference on information processing, ICIP, Springer Verlag Berlin Heidelberg, CCIS 157, pp 287–293
[22] Jadhav AS, Sonar RM (2011) Framework for evaluation and selection of the software packages: a hybrid knowledge based system approach. J SystSoftw 84(8):1394–1407
[23] Bell DE, La Padula LJ (1973) Secure computer systems: Vol. I— mathematical foundations, Vol. II—a mathematical model, Vol. III—a refinement of the mathematical model. Technical Report MTR-2547 (three volumes), Mitre Corporation, Bedford, MA, March–December
[24] Anderson RJ (1996) Security engineering, Wiley
[25] Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48:203–209
[26] Koblitz N (1989) Hyperelliptic cryptosystems. J Cryptol 1(3):139–150
[27] Stallings W (2003) Cryptography and network security principles and practices, 3rd edn. Pearson Education, Upper Saddle River
[28] Dimitriadis CK (2007) Analyzing the security of internet banking authentication mechanisms. Information systems control journal 3:1–8
[29] Sommerville I (2003) Software engineering. Pearson Education, London. ISBN 8129708671
[30] Common Criteria for Information Technology Security Evaluation (2006) Version 3.1 http://www.Commoncriteriaportal.Org/ public/expert
[31] Kotonya G, Sommerville I (1995) Requirement engineering with viewpoints
[32] Schneier B (1996) Applied cryptography, Wiley
[33] Hiltgen A, Kramp T, Weigold T (2005) Secure internet banking authentication. IEEE Securpriv 1540(7993):24–32
[34] Hankerson Darrel, Menezes Alfred, Vanstone Scott (2004) Guide to elliptic curve cryptography. Springer, New York. ISBN 0-387- 95273-X
[35] M. Schiffman, "Common Vulnerability Scoring System (CVSS)," http://www.first.org/cvss/
[36] Assad Ali, PavolZavarsky, Dale Lindskog, and Ron Ruhl, " A Software Application to Analyze Affects of Temporal and Environmental Metrics on Overall CVSS v2 Score", Concordia University College of Alberta, Edmonton, Canada, October 2010.
[37] National Vulnerability Database 2014.
[38] R. Ortalo, Y. Deswarte, and M. Kaaniche, "Experimenting with quantitative evaluation tools for monitoring operational security," IEEE Transactions on Software Engineering, vol. 25, pp. 633–650, September 1999.
[39] W. Li and R. Vaughn, "Cluster security research involving the modeling of network exploitations using exploitation graphs," in Sixth IEEE International Symposium on Cluster Computing and Grid Workshops, May 2006.