

Security Architecture Modeling & Design Using Rule Based Decision Tree Approach

A Dissertation Submitted in partial fulfillment of the requirement

For the Award of degree of

MASTER OF ENGINEERING

In

Computer Technology and Application

By

Rahul Thukral

(11/CTA/08)

(University Roll No. 8409)

Under the Guidance of

Dr. DAYA GUPTA



DEPARTMENT OF COMPUTER ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

2008-2010

CERTIFICATE



DELHI TECHNOLOGICAL UNIVERSITY

(Govt. of National Capital Territory of Delhi)

BAWANA ROAD. DELHI - 110042

This is to certify that project entitled “**Security Architecture Modeling & Design using Rule Based Decision Tree Approach**” has been completed by **Mr. Rahul Thukral , University Roll No. 8409** in the partial fulfillment of the requirement for the award of degree of **Master in Engineering in Computer Technology & Application.**

This is a bonafied work carried out by him under my supervision and support. This is a beneficial work in field of security engineering and cryptography.

(Dr. DAYA GUPTA)

HOD & PROJECT GUIDE

(Dept. of Computer Engineering)

DELHI TECHNOLOGICAL UNIVERSITY

ACKNOWLEDGEMENT

This work is not a solo endeavor but rather the amalgamate consequence of contribution from various people and sources. Therefore it would be discourteous to present it without acknowledging their valuable guidance.

It is distinct pleasure to express my deep sense of gratitude and indebtedness to my learned supervisors Dr. Daya Gupta (Head of Computer Engineering Dept, Delhi Technological University) for their invaluable guidance, encouragement and patient reviews. Their continuous inspiration only has made me complete this dissertation. All of them kept on boosting me time and again for putting an extra ounce of effort to realize this work.

I would also like to take this opportunity to present my sincere regards to Mrs. Kakali Chatterjee for her technical assistance and invaluable suggestions.

Finally we are also thankful to my classmates for their unconditional support and motivation during this work.

(Rahul Thukral)

Master in Engineering

(Computer Technology & Application)

Dept. of Computer Technology

DELHI TECHNOLOGICAL UNIVERSITY



Delhi Technological University

ABSTRACT OF THE MASTER'S THESIS

Author :	Rahul Thukral
Name of the Thesis :	Security Architecture Modeling & Design Using Rule Based Decision Tree Approach
Department :	Computer Engineering
Supervisor :	Dr. Daya Gupta , Head of Dept , Computer Engineering
<p>The absence of security discipline in today's software development practices often produces software with exploitable weaknesses. The ubiquitous dependence on information technology makes software security a key element of business continuity, disaster recovery, incident response, and national security.</p> <p>Adopting a security-enhanced software development process that includes secure development practices will reduce the number of exploitable faults and weaknesses in the deployed software. Correcting potential vulnerabilities as early as possible in the SDLC, mainly through the adoption of security-enhanced processes and practices, is far more cost-effective than attempting to diagnose and correct such problems after the system goes into production.</p> <p>So, we propose a Model framework for converting Security requirements & threats, which are identified during the security requirement elicitation stage, into design decisions and guidelines. The process we will describe will be called as SDT (Security Design Template) which is based upon a set of important decisive security attributes that influences design choices at the various layers of security engineering. We propose to use the Decision Tree approach, to judge the best suitable security protocol that will help mitigate the identified security requirement or threats.</p>	
<p>Keywords : Security Engineering , Security Requirement Engineering , Security Requirement Elicitation , Prioritization , Security Architecture & Design Engineering, Threats, Cryptosystems , Security Design Template , Decision Trees</p>	

Contents

CHAPTER 1 Introduction To Security Engineering	9
1.1 General Concept	9
1.2 Security Engineering.....	10
1.3 Motivation	13
1.4 Proposed Work.....	14
1.5 Outline	16
 CHAPTER 2 Security Engineering Process (SEP)	 17
2.1 Security	17
2.2 Security Engineering.....	19
2.2.1 View on Security Engineering	19
2.3 Security Engineering Process (SEP)	21
 CHAPTER 3 Security Requirement Engineering	 25
3.1 Requirement Engineering	25
3.2 Security Requirements.....	25
3.2.1 Identification Requirement.....	27
3.2.2 Authentication Requirement	27
3.2.3 Authorization Requirement	27
3.2.4 Immunity Requirement.....	28
3.2.5 Integrity Requirement.....	28
3.2.6 Intrusion Detection Requirement.....	29
3.2.7 Non Repudiation Requirement.....	29
3.2.8 Privacy Requirement.....	30
3.2.9 Security Auditing Requirement	30
3.2.10 Survivability Requirement	31
3.2.11 System Maintenance Requirement	31
3.3 Security Requirement Elicitation	32
3.3.1 Security Requirement Discovery & Definition	40
3.4 Security Requirement Analysis & Prioritization	45
3.4.1 Analyzing Security Requirements	45
3.4.2 Security Requirement Prioritization	46
3.4.2.1 Evaluation Of Threats	46
3.4.2.2 Prioritizing Security Requirements	49
3.4.3 Managing Security Requirements	51
3.5 Summary.....	53

CHAPTER 4 Security Engineering & Cryptography.....	54
4.1 Cryptographic Engineering.....	55
4.2 Cryptography Services.....	57
4.3 Security Attacks.....	58
4.3.1 Cryptographic Attack Methods.....	58
4.3.2 Implementation Specific Attacks.....	61
4.4 Analysis Of Strength Of Various Cryptography Protocols.....	62
4.4.1 Metrics for Characteristics of Cryptography Protocol	62
4.4.3 Application	64
4.4.3.1 Data Encryption Standard / Data Encryption Algorithm	64
4.4.3.2 3DES (Triple DES).....	65
4.4.3.3 RC5.....	66
4.4.3.4 RSA.....	67
4.4.3.5 ECC.....	67
4.4.3.6 HECC	69
4.5 Comparision	70
CHAPTER 5 Security Design Engineering.....	72
5.1 Security Design Engineering.....	73
5.1.1 Identification Of Cryptography Services.....	75
5.1.2 Security Design Structuring.....	76
5.1.2.1 Identify Security Design Attributes	76
5.1.2.2 Mapping through Security Design Template.....	78
5.1.3 Security Design Decisions	80
CHAPTER 6 Case Study for ‘ Advanced Health Care System ‘	82
6.1 Requirement Elicitation & Priortization	82
6.2 Security Design Phase	92
6.3 Example for An ATM System	97
CHAPTER 7 Conclusion	103
References	104

List of Figures

Figure 1 : Core security properties of secure system	18
Figure 2 : The Security Engineering Process (SEP)-A Modified version of Conventional Spiral Model .	22
Figure 3 : Different types of stake holders as according to view point	41
Figure 4 : Process for Threat Priority	47
Figure 5 : Security Requirement Prioritization	49
Figure 6 : Model for Managing Security Requirements	51
Figure 7 : Framework for Security Engineering Process (SEP) Highlighting Design Phase	74
Figure 8 : Mapping Cryptography Services to Security Requirements	75
Figure 9 : Decision Tree	81
Figure 10 : Viewpoint Hierarchy for Advanced Medical Care System	83
Figure 11 : Mapping Security Requirements to Cryptography Services	92
Figure 12 : Decision Tree Path for Privacy Requirement	95
Figure 13 : Security design Decisions	96
Figure 14 : Abuse Case for Specialist Doctor in Advanced Health Care System	33
Figure 15 : Abuse Case for Hospital Staff in Advanced Health Care System	33
Figure 16 : Misuse Case for Advanced Health Care System	36
Figure 17 : Security Use Case for Advanced Health Care System	37
Figure 18 : Attack Tree for Advanced Health Care System	38
Figure 19 : Mapping Security Requirements to Cryptography Services for ATM System	98
Figure 20 : Security design Decisions for Privacy requirement of ATM System	102
Figure 21 : Decision Tree for Design Decisions for ATM System	101

List of Tables

Table 1 :	Specialist Doctor Profile for Access Patient Reports	42
Table 2 :	Threats Category and description	43
Table 3 :	Example of “Advanced Medical Care System” using VOSREP	44
Table 4 :	Measure of Vulnerability	48
Table 5 :	Three – Dimension lookup table to measure the level of risk	49
Table 6 :	Cryptography Services & Protocols	71
Table 7 :	List of Design Attributes	76
Table 8 :	Values Categories of Design Attributes.....	78
Table 9 :	Values of Design Attributes	79
Table 10 :	Security design template for (T.Change_Data , Patient Records ,Privacy Requirement) ..	79
Table 11 :	Specialist Doctor profile for Access Patient Reports	85
Table 12 :	Example Showing evaluation of threat based upon Stakeholders Profile.....	86
Table 13 :	Example of “Advanced Medical Care System” using VOSREP	87
Table 14 :	Possible Vulnerable Assets	88
Table 15 :	Measure of Asset	89
Table 16 :	Measure of Threat	89
Table 17 :	Level of Vulnerability	90
Table 18 :	Measurement of Risk Levels.....	91
Table 19 :	Security Requirement , Threat , Asset , Risk Table	92
Table 20 :	Mapping Cryptography Services to Viewpoint , Asset , Threat & Security Requirement ..	93
Table 21 :	Design Attributes	93
Table 22 :	Values of Design Attributes for Threat T.Change_Data on Asset Patient Records	94
Table 23 :	Security design template for (T.Change_Data , Patient Records , Privacy Requirement) ..	94
Table 24 :	Comparable Key Sizes in Terms of Computational Effort for Cryptanalysis	70
Table 25 :	Threats & Actor	35
Table 26 :	One of the Identified Security requirement for an ATM System	97
Table 27 :	Mapping Cryptography Services to Viewpoint , Asset for ATM System	98
Table 28 :	Values of Design Attributes for Threat T.Data_Theft on Credit Card Info	99
Table 28 :	SDT for (T.Data_Theft,ATM/Debit/Credit Card Info,Privacy Req)	100

CHAPTER 1 Introduction to Security Engineering

Software is ubiquitous. Many of the products, services, and processes that organizations use and offer are highly dependent on software to handle the sensitive and high-value data on which people's privacy, livelihoods, and very lives depend.

This ubiquitous dependence on information technology makes software security a key element of business continuity, disaster recovery, incident response, and national security. Software vulnerabilities can jeopardize intellectual property, consumer trust, business operations and services, and a broad spectrum of critical applications and infrastructures, including everything from process control systems to commercial application products.

The absence of security discipline in today's software development practices often produces software with exploitable weaknesses. Security-enhanced processes and practices—and the skilled people to manage them and perform them—are required to build software that can be trusted to operate more securely than software being used today.

1.1 General Concept

Security of software system [1] *is defined as technological and managerial procedures applied to computer systems to ensure the CIA (confidentiality, integrity and availability) of information managed by the system.* In other words security of software system means the protection afforded to an automated information system in order to attain the applicable objectives of preserving the CIA (Confidentiality, integrity and availability) of information system resources that includes hardware, software, firmware, information/data, and telecommunications.

Organizations increasingly store, process, and transmit their most sensitive information using software-intensive systems that are directly connected to the Internet. Private citizens' financial transactions are exposed via the Internet by software used to shop, bank, pay taxes, buy insurance, invest, register children for school, and join various organizations and social networks. The increased exposure that comes with global connectivity has made sensitive

information and the software systems that handle it more vulnerable to unintentional and unauthorized use. In short, software-intensive systems and other software-enabled capabilities have provided more open, widespread access to sensitive information—including personal identities—than ever before.

The number of threats specifically targeting software is increasing, and the majority of network- and system-level attacks now exploit vulnerabilities in application-level software. According to CERT analysts at Carnegie Mellon University[2] most successful attacks result from targeting and exploiting known, unpatched software vulnerabilities and insecure software configurations, a significant number of which are introduced during software design and development.

These conditions contribute to the increased risks associated with software-enabled capabilities and exacerbate the threat of attack. Given this atmosphere of uncertainty, a broad range of stakeholders need justifiable confidence that the software that enables their core business operations can be trusted to perform as intended.

1.2 Security Engineering

Security engineering is about building systems that remain dependable in the face of malice, error, or mischance. As a discipline, it focuses on the tools, process, and methods needed to design, implement, and test complete systems, and to adapt existing systems as their environment evolves. Security engineering indeed requires cross-disciplinary expertise, ranging from cryptography and computer security through system engineering skills, from the business process analysis through software engineering, to evaluate and test the system. So we define security engineering as -

A security engineering process is a complex activity involving many special work products such as security requirement elicitation, prioritization, security design, and implementation and testing. These work products are essential in a process that aims to create trustworthy information security products [12].

Security engineering entails using practices, processes, tools, and techniques to address security issues in every phase of the software development life cycle (SDLC). Software that is developed with security in mind is typically more resistant to both intentional attack and unintentional failures. One view of secure software is software that is engineered "so that it continues to function correctly under malicious attack" and is able to recognize, resist, tolerate, and recover from events that intentionally threaten its dependability. Broader views that can overlap with software security (for example, software safety, reliability, and fault tolerance) include the notion of proper functioning in the face of unintentional failures or accidents and inadvertent misuse and abuse, as well as reducing software defects and weaknesses to the greatest extent possible regardless of their cause.

The advantage of using security engineering process is to build better and defect-free systems. Software-intensive systems that are constructed using more securely developed software are better able to do the following:

- Continue operating correctly in the presence of most attacks by either resisting the exploitation of weaknesses in the system by attackers or tolerating the failures that result from such exploits
- Limit the damage resulting from any failures caused by attack-triggered faults that the system was unable to resist or tolerate and recover as quickly as possible from those failures

The objective is to increase the security and dependability of the system, produced by these practices, both during its development and during its operation. Thus, from the above facts we can say that *security engineering is becoming essential component of systems engineering*.

So, to design, build, and deploy secure systems, we must integrate security into your application development life cycle and adapt your current software engineering practices and methodologies to include specific security-related activities. Security-related activities include [3] identifying security requirements, prioritizing security requirements , applying security

design, Implementing security mechanisms , security testing, and conducting security deployment reviews.

The different activities of Security Engineering Process (SEP) can be categorized into following phases –

- **Requirement Engineering** – Discover security requirement along with functional and non functional requirements such as Privacy, Authentication, Integrity, Non-Repudiation requirements, elicit and prioritize them.
- **Design Decisions** – With true security requirements specified most appropriate design decisions can be taken. The different activities taken in security design includes identifying cryptography services & security design attributes , structuring them with threat and asset and finally taking design decision that specifies which security protocol is best suited for the identified Security Requirement.
- **Implementation** – This includes implementing specific algorithms that are suggested in the design phase of the Security Engineering Process.
- **Testing** - It involves evaluating the system security and determining the adequacy of security mechanisms, assurances and other properties to enforce system security policies .The primary reason for testing the security of an operational system , is to uncover design , implementation and operational flaws that could allow the violation of system security And to find unidentified potential vulnerabilities and subsequently repair them before delivering the final system to the user. The identified design decisions are validated against the security requirements and the extent to which they satisfy a particular security requirement.

1.3 Motivation

A lot of work has been reported on discovering and eliciting of security requirements like abuse case [12], misuse [13], common criteria [14,15], attack trees [16] and VOSREP [6] that proposes seeming less integration of security requirement elicitation activities in conventional requirement engineering process. There is also Risk Prioritization and Management methods like EBIOS [17], OCTAVE [18], CORAS [19], and CRAMM [20] which enforces security levels based on risk measure.

Once the security requirements are elicited and prioritized, if proper Design Decisions are not taken, then it can lead to an underdeveloped system with unnecessary design constraints which makes the application vulnerable & exposed to attackers during its operation. Attacks may take advantage of publicly known but unpatched vulnerabilities, leading to memory corruption, execution of arbitrary exploit scripts, remote code execution, and buffer overflows. Software flaws can be exploited to install spyware, adware, and other malware on users' systems that can lie dormant until it is triggered to execute [8].

The Design Phase of Security Engineering process has not received sufficient standardization and work in the recent past. Some recent work in methodology for security policy definition using the Zachman information systems architecture [21] has been proposed. In fact, it is widely recognized that most of the threats in real-world security infrastructure stems from how we perform cryptographic operations on secret data, although only a subset of security threats relating to privacy, authentication, integrity and non-repudiation services would be mitigated through the use of cryptography protocols. As a consequence, the design details, which normally take on a marginal role and seemingly just affect performance, are of crucial importance, as they could open door to many real-world attacks in a number of nontrivial and often unforeseen ways.

Hence we must have techniques that focus on converting identified security requirements into design decision that mitigates the identified security threat. To do this, first we identify cryptography services & design attributes, then design structuring that involves mapping

Threats , Asset & Identified security requirement with the design attributes , and at the end design decision are taken that tells which security protocol is best suited to eliminate the identified security threats.

*Therefore we aim to develop a well defined Framework for Security Engineering Process (SEP) highlighting the design process, that will have well articulated steps for **Security Design Engineering**. Moreover this process should be coherent with the conventional Software Engineering process so that eliciting security requirements & security design become an integral part of system engineering and security engineering.*

1.4 Proposed Work

In this thesis work, we propose a Framework for Security Engineering Process (SEP) highlighting the design phase, that involves converting Security requirements & threats, which are identified during the security requirement elicitation stage, into design decisions and guidelines. The process we will describe will be called as SDT (Security Design Template) which is based upon a set of important decisive security attributes that influences design choices at the various layers of security engineering process. We propose to use The Decision Tree approach, to judge the best suitable security protocol that will help mitigate the identified security requirement or threats.

An extension of the well defined spiral process model of SDLC is also proposed, that integrates Security requirement & design phases into the application development life cycle and adapt the current software engineering practices and methodologies to include specific security-related activities & depicting how cryptographic engineering requirements influence design choices at the various layers of Security Engineering Process.

In this thesis we shall focus on the Security Design Engineering and shall be presenting techniques for –

- **Identification of Cryptography Services** – After the security requirements have been elicited and prioritized in the requirement phase, the different types of security requirements are mapped to the different security services provided by cryptography like Confidentiality services, Integrity services, Authentication services and Non-repudiation services. The identified cryptography services would eventually help in the later stages of the design process, by specifying which cryptography techniques would be usefully in a particular scenario.
- **Security Design Structuring** – *In this activity the design attributes are identified that affects the design decisions .These design attributes are chosen based upon the different decisive attributes of the cryptography protocols that best satisfies the security requirements of the system under study.A security design template (SDT) is also prepared for each and every threat and asset pair identified in the requirement phase. A Security Design Template is a collection of chosen design attributes which are validated with the environmental constraints specified in the requirement phase.*
- **Security Design Decision** *In this activity , Based upon the extensive Literature Review & Comparison of the efficiency of various security protocols , we propose a rule based decision tree approach , to judge the best suitable security protocol depending upon the design attributes supplied in the ‘Design Template’ .*

The advantage of using this approach for security engineering helps in the identification of true security requirements & design guidelines. With true security requirements have been identified, systematically analyzed and specified the architecture team can choose most appropriate security mechanisms to implement them and thus making the system under development to be more efficient, reliable and secure.

1.5 Outline

This aim of this dissertation is to provide a framework for security engineering process that will elicit security requirements along with functional and non-functional requirements, prioritize them and convert them into design decision . The approach if used for development of software systems results in the systems that are less vulnerable, cost effective and secure. The rest of the thesis is organized as follows.

Chapter 2 gives an overview of the *Security Engineering Process (SEP)* along with a description of different phases of the SEP Process.

Chapter 3 addresses what is Requirement Engineering, what are security requirement and different types of *Security Requirements* for any computer based systems, view point oriented requirement definition, different security requirement elicitation techniques and different risk management techniques.

Chapter 4 addresses the role of *Cryptography in Security Engineering* and gives a comparison of various cryptography protocols based on their key attributes. It discusses from the security engineering perspective, cryptography forms an important factor in the design considerations of building a secure system.

Chapter 5 discusses the various activities that are involved in the *Security Design Engineering* including Identification of Cryptography Services , Design Structuring, and finally Design decisions.

Chapter 6 presents a Case study for ‘Advanced Health Care Systems ‘based upon the proposed methods.

Chapter 7 concludes the Thesis.

CHAPTER 2 Security Engineering Process (SEP)

2.1 Security

Security is an attribute of system that prevents the system from revealing, changing and denying of resource services and system information in an illegal way. Generally three aspects of security are: confidentiality, integrity and availability of service of resources and information. To achieve these aspects and develop a secure system, security services and mechanisms should be considered.

- [Confidentiality](#). The software must ensure that any of its characteristics (including its relationships with its execution environment and its users), its managed assets, and/or its content are obscured or hidden from unauthorized entities. This remains appropriate for cases such as open-source software; its characteristics and content are available to the public (authorized entities in this case), yet it still must maintain confidentiality of its managed assets.
- [Integrity](#). The software and its managed assets must be resistant and resilient to subversion. Subversion is achieved through unauthorized modifications to the software , managed assets, configuration, or behavior by authorized entities, or any modifications by unauthorized entities. Such modifications may include overwriting, corruption, tampering, destruction, insertion of unintended (including malicious) logic, or deletion. Integrity must be preserved both during the software's development and during its execution.
- [Availability](#). The software must be operational and accessible to its intended, authorized users (humans and processes) whenever it is needed. At the same time, its functionality and privileges must be inaccessible to unauthorized users (humans and processes) at all times.

Two additional properties commonly associated with human users are required in software entities that act as users:

- [Accountability](#). All security-relevant actions of the software-as-user must be recorded and tracked, with attribution of responsibility. This tracking must be possible both while and after the recorded actions occur. The audit-related language in the security policy for the software system should indicate which actions are considered "security relevant."
- [Non-repudiation](#). This property pertains to the ability to prevent the software-as-user from disproving or denying responsibility for actions it has performed. It ensures that the accountability property cannot be subverted or circumvented.



Figure 1. Core security properties of secure system

2.2 Security Engineering

Security engineering is defined by “a set of methodologies and technologies for fast and cheap development and operation of high quality secure systems by means of applying cryptology, information security technology and software engineering.” Security engineering covers all phases of life-cycle, those are requirement analysis, design, implementation, testing, maintenance phase, of development of a secure system.

2.2.1 View on security engineering

(1) Common objects of engineering

Most of engineering subjects, including security engineering, electrical, mechanical, civil, chemical, system and bio-medical engineering, have the following common objects:

- *Standard and assurance*: All material, process, product, quality should be standardized by standard organizations (e.g., ISO/IEC, ANSI, BS, KS, JIS) and be assured by evaluation and certification authority. For example, in security engineering, cryptographic algorithms (e.g., DES, AES, and so on) and cryptographic protocols (e.g., SET, SSL), should be standardized and assured.
- *Quantification*: Development and maintenance cost and time, quality of product and man-month should be quantifiably measured. Thus they can be controlled and managed. In security engineering, evaluation assurance level, risk level, reliability and security strength should be quantifiably measured.
- *Cost-effective*: We should maximize the return of interest which is “the principle of economics”.
- *User centric*: Output or product should be useful for end user.
- *Effectiveness*: All solutions should be practical and feasible, even they are not optimal solution.

- Documentation: All material, process, product, quality should be formally documented.

(2) Relating technology

Security engineering technology has the following relating technologies:

- Information security technology: cryptography, cryptographic protocol, security service (e.g., nonrepudiation, authentication, access control, and so on) and conventional information security technology
- Software engineering technology: architecture technology for cryptographic object or component
- Security evaluation technology: information security system evaluation (or assessment) and authentication technology
- Security management technology: organization's security management technology, security policy technology, risk evaluation technology, and so on. Additionally there are two approaches on security engineering and software engineering
- Security engineering for software: Security engineering and cryptographic technologies are used for the purpose of protecting source code of software. Copyright protection technology such as digital watermark, DRM technology, and so on is the example.
- Software engineering for security: It is narrow view on security engineering. Conventional software technologies are applied to security engineering. Note that security engineering is an instance of software engineering.

(3) Principles on security engineering

- Minimization of development *cost* and development *duration* in a security product development.

- Maximization of *quality* (security, usability, maintainability, and so on) of a developed security product.
- Providing just as much as needed assurance level and security function in security system construction.
- Obtain maximum security strength by using minimum cost in security system construction.

2.3 Security Engineering Process (SEP)

To design, build, and deploy secure systems, we must integrate security into your application development life cycle and adapt your current software engineering practices and methodologies to include specific security-related activities. Security-related activities include [3] identifying security requirements, prioritizing & management of security requirements , security design , Implementing security mechanisms , security testing, and conducting security deployment reviews.

As in the conventional spiral model the process starts from the requirement discovery & definition where the developer will meet the client and collect all the information related to project development then analyze them and check feasibility of the project and if it is feasible, then he will proceed and plan the further phases of the development, and follow all the successive steps accordingly. *But in this process there is no consideration of security requirement in the early phase that is the requirement engineering phase which is the first phase.* It will consider all the constraint related to security at the end of development process. So handling of these at later phase will cost more and will sometime lead to over budgeting or failure of project.

So it will be better to incorporate security related activities at all the stages of the software development lifecycle. So we have modified the conventional spiral model to incorporate security engineering phases in it.

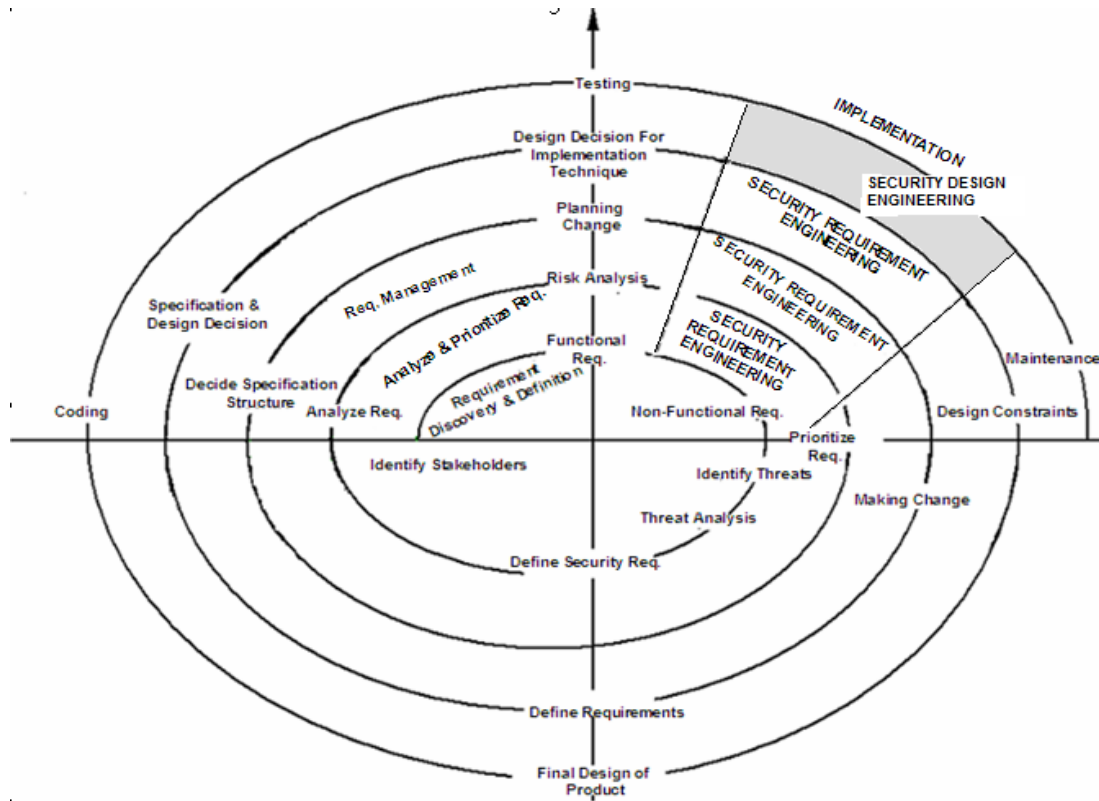


Figure 2 – The Security Engineering Process (SEP) 'A Modified version of Conventional Spiral Model'

All the activities defined in the forgoing section are spiral in nature (Figure 2). The merit of doing these activities in spiral is that one has not to wait for the first activity to finish rather he can start the other activities in the process this helps in the development of the product from scratch or it can also be used for the enhancement of the existing systems.

The first three spirals that are shown in Figure 2 shows the three activities of the process described above as shown along the radial dimension of second quadrant. After our process is completed, means that we have elicited, analyzed, prioritized and managed all the security requirements of the system under question. Once this process is done the design team can take the most appropriate design decisions for mitigating the threats which were identified during the security requirement phase. This is shown along the fourth spiral of the figure shown above. Once the design decisions are taken for different security requirements the implementation team can write the appropriate code so that the system under development is completed fully as shown in last or fifth spiral of the Figure 2.

There are a number of distinct security-related activities that should be an integral part of your application life cycle. These different activities of Security Engineering Process (SEP) can be categorized into following phases –

- **Requirement Engineering** – It involves discovering security requirement along with functional and non functional requirements such as Privacy, Authentication, Integrity, Non-Repudiation requirements, elicit, prioritize and manage them.
- **Design Decisions** – With true security requirements specified most appropriate design decisions can be taken. The different activities taken in security design includes identifying security design attributes , structuring them with threat and asset and finally taking design decision that specifies which security protocol is best suited for the identified Security Requirement.
- **Implementation** – This includes implementing specific algorithms that are suggested in the design phase of the Security Engineering Process.
- **Testing** - It involves evaluating the system security and determining the adequacy of security mechanisms, assurances and other properties to enforce system security policies .The primary reason for testing the security of an operational system , is to uncover design , implementation and operational flaws that could allow the violation of system security And to find unidentified potential vulnerabilities and subsequently repair them before delivering the final system to the user. The identified design decisions are validated against the security requirements and the extent to which they satisfy a particular security requirement.

In the first phase of security engineering process, first of all, different stakeholders must be identified as shown at the start of spiral then we will define functional and non- functional requirements. As we have functionalities we can identify various possible threats to the functionalities, these threats help in eliciting security requirements. Finally these security

requirements are prioritized based on risk measure in requirement engineering phase. This first phase has been discussed in detail in chapter 3.

Our Thesis focuses on the second phase of SEP i.e. the Design Phase. With true security requirements identified, systematically analyzed and specified, the design team can choose the most appropriate security mechanisms to implement & thus making the system under development to be more efficient, reliable and secure. This phase will be discussed in detail in chapter 5.

CHAPTER 3 Security Requirement Engineering

3.1 Requirements Engineering

It is the process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed. The requirements are nothing but are the descriptions of the system services and constraints that are generated during the requirements engineering process.

“Requirements as defined by Davis are the high level abstraction of the service or Constraint of a system.”

The different types of requirement as described by sommerville are as follows:-

- Functional Requirements.
- Non Functional Requirements.
- Domain Requirements.
- Safety Requirements.
- Security Requirements.

So, In this chapter we focus on how security requirement can be engineered i.e. The Security Requirement Engineering Process.

3.2 Security Requirements

“Security Requirements is defined as a high level requirement that gives detail specification of the system behavior that is unacceptable such as all users’ application can only access data for which they are properly authorized. They differ from safety requirements which are domain specific and more suitable for control systems application. They are also known as shall not requirements but are not risks or threats”.

These requirements are very important for successful operation of the system. It is defined as the process of identification of requirements in parallel to other requirements that are

functional, non-functional and domain requirements. If these requirements are not properly elicited, analyzed and managed they result in a system that can fail. So they are even more important than all the requirements described above. These requirements are generally called as security requirements since they are responsible for the security of the system. The process of eliciting, analyzing and managing these security requirements is called as **security requirements**.

There are some major differences between security requirements and the requirements that are described in the above section. These differences are listed below –

- Security requirements are different from functional requirements which are derived from goals of system whereas security requirements are objective resulting from threats on functionality or confidential data.
- Security requirements are related to non functional requirements such as correctness, interoperability, feasibility etc. For example non functional requirement such as correctness, if implemented covers to some extent the integrity security requirement.
- Security requirements are also different from architectural constraint because these constraints unnecessarily prevent architecture team from using efficient mechanism to satisfy needed security requirements.

Security requirements are also different from Safety requirements in following respects –

- As security requirements are having well defined security life cycle and various standards whereas a safety requirement does not have;
- Security requirements are having generic threats rather than system specific hazards
- Mature security technology (encryption, etc.) as compared to safety requirements.

Different types of security requirements as proposed by Firesmith [23] are as follows –

2.2.1 Identification Requirement

Identification requirement specifies the extent to which a CBS shall identify its external environment.

Examples –

- The main application shall identify all its client applications, human users before allowing them to use its capabilities.
- All persons should be identified before allowing them to enter.

2.2.2 Authentication Requirement

It is the security requirement that specifies that CBS should verify the identity of its externals. The typical objective of this security requirement is to ensure that externals are actually who or what they claim to be.

Examples –

- Application shall verify the identity of all of its users before allowing them to do any interaction (message, transaction) with the system.
- Before permitting the personnel to interact with data center there identities should be verified.

2.2.3 Authorization Requirement

This security requirement specifies that only authenticated externals can access specific application capabilities or information only if they have been explicitly authorized to do so by the administrator of the application.

Examples –

- The application shall allow the customer to obtain access to his/her account information rather than of other customer.
- Application shall not allow intruders access the credit card information of customers.
- Application shall not allow users to flood the system.

2.2.4 Immunity Requirement

An immunity requirement is any security requirement that specifies an application shall protect itself from infection by unauthorized undesirable programs (e.g., computer viruses, worms, and Trojans).

Examples –

- Application shall protect itself from infection by scanning data for viruses, worms, Trojan, and other harmful programs
- Application shall delete or disinfect the file found to be infected.
- Application shall notify the user if it detects a harmful program.

2.2.5 Integrity Requirement

This security requirement specifies ensures that its data does not get corrupted via unauthorized creation, deletion, modification.

Examples –

- The application shall prevent the unauthorized corruption of emails that it sends to customers.

- The application shall prevent the unauthorized corruption of data collected from customers and other external users.
- The application shall prevent the unauthorized corruption of all communications passing through networks.

2.2.6 Intrusion detection Requirements

This security requirement specifies that if an application has been attacked by intruders then that can be detected and recorded so that the administrator can handle them.

Examples –

- The application shall detect and record all attempted accesses that fail identification, authentication, or authorization requirements.
- The application shall notify the security officer of all failed attempted accesses.

2.2.7 Non repudiation requirements

This security requirement specifies that a party should not deny after interacting (e.g. message, transaction) with all or part of the interaction.

Examples –

The application shall make and store records of the following information about each order received from a customer and each invoice sent to a customer –

- The contents of the order or invoice.
- The date and time that the order or invoice was sent.
- The date and time that the order or invoice was received.

- The identity of the customer.

2.2.8 Privacy Requirements

This security requirement specifies that the application should keep its data and communications private from unauthorized individuals and programs.

Examples –

- Anonymity Privacy – The application shall not store any personal information about the users.
- Communication Privacy – The application shall not allow unauthorized individuals or programs access to any communications.
- Data Storage Privacy – The application shall not allow unauthorized individuals or programs access to any stored data.

2.2.9 Security Auditing Requirements

A security auditing requirement specifies that an application shall enable security personnel to audit the status and use of its security mechanisms.

Examples –

The application shall collect, organize, summarize, and regularly report the status of its security mechanisms including –

- Identification, Authentication, and Authorization.
- Immunity
- Privacy
- Intrusion Detection

2.2.10 Survivability Requirements

The security requirement specifies that that an application should work possibly in degraded mode even if some destruction has been there in the application.

Examples –

- The application shall not have a single point of failure.
- The application shall continue to function (in degraded mode) even if a data center is destroyed.

2.2.11 System Maintenance requirements

This requirement specifies that how the modifications can be done so that security fixes that have been detected can be resolved.

Examples –

- The application shall not violate its security requirements as a result of the upgrading of a data, hardware, or software component.
- The application shall not violate its security requirements as a result of the replacement of a data, hardware, or software component.

3.3 Security Requirement Elicitation

Using an elicitation method can help in producing a consistent and complete set of security requirements. However, brainstorming and elicitation methods used for ordinary functional (end-user) requirements usually are not oriented toward security requirements and, therefore, do not result in a consistent and complete set of security requirements. The resulting system is likely to have fewer security exposures when requirements are elicited in a systematic way.

The following list identifies several methods that could be considered for eliciting security requirements. Some have been developed specifically with security in mind (e.g., misuse cases), whereas others have been used for traditional requirements engineering and could potentially be extended to security requirements.

- Misuse cases [24,13]
- Abuse Case [12]
- Common Criteria [14,15]
- Attack Trees [16]
- VOSREP(View point Oriented Security Requirement Elicitation Process) [6]
- Soft Systems Methodology [25]
- Quality Function Deployment [26]
- Controlled Requirements Expression [27]
- Issue-based information systems [28]
- Critical discourse analysis [29]

Abuse Cases

Abuse case [12] is a specification of complete interaction between a system and one or more actors, where the interaction can cause harm. A complete abuse case defines an interaction between an actor and the system that results in harm to a resource associated with one of the actors, one of the stakeholders, or the system itself. A further distinction we make is that an abuse case should describe the abuse of privilege used to complete the abuse case. Clearly, any abuse can be accomplished by gaining total control of the target machine through modification

of system software or firmware. Abuse cases can be described using the same strategy as for use cases. We distinguish the two by keeping them separate and labeling the diagrams. Figure 14 , 15 are showing the concept with the example Advanced Heath Care System.

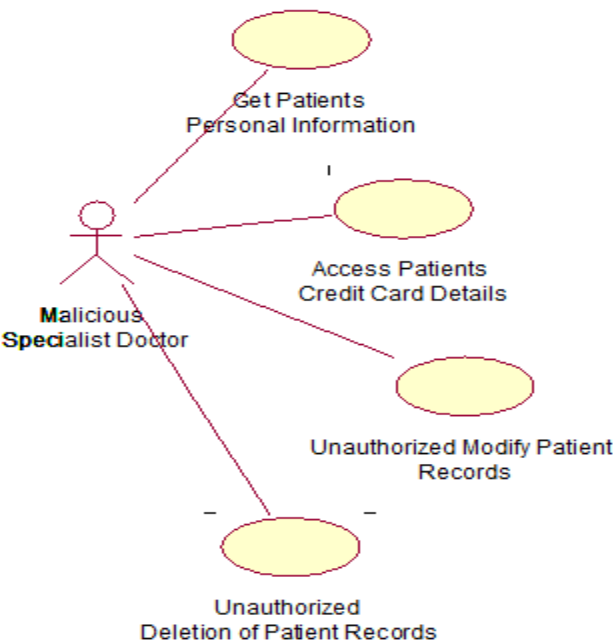


Figure 14 : Abuse case for Specialist Doctor in Advanced Health Care System

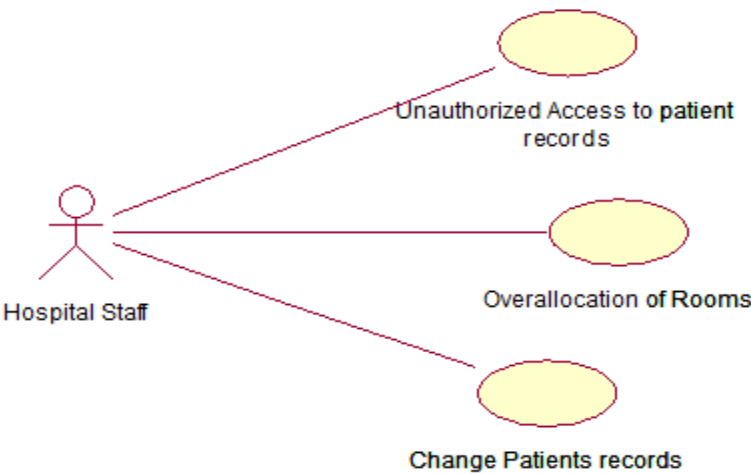


Figure 15 : Abuse case for Hospital Staff in Advanced Heath Care System

Common Criteria (CC) with use cases –

This approach specifies how standards such as common criteria [14, 15] can be correlated with use case diagrams. The purpose of correlating use case and common criteria is to handle security in IT products during the software engineering process itself.

It has following steps:

- For the Purpose of correlating common criteria with use case diagrams the approach makes it mandatory to complete the actor profiles for each actor involved in the use case diagram.
- Actor profile has seven fields consisting of:
 - Its name
 - Functionality
 - Type of Actor that may be
 - Human
 - Corporative
 - Autonomous
 - Location
 - Local
 - Remote
 - Use Case Association
 - Read
 - Write
 - Read_write
 - Ask
 - Answer
 - Ask_answer
- Weather or not the use case involves exchanging private information
- Weather or not the use case involves secret information exchange.

- After the use case creator completes the actor profiles, these actor profiles are used to maps vulnerable threats to the actor from a predefined set of threat categories. As it has maintained threat repository so we can get threats by completing the threat profile as shown in Table 25. Now these threats are used to find out the security requirements.

Association = read	Association = write
Impersonate	Change_Data
Repudiate_Receive	Repudiate_Receive
If(Private Exchange = true) Privacy_Violated	If(Private Exchange = true) Privacy_Violated
If(Secret Exchange = true) Data_Theft	If(Secret Exchange = true) Data_Theft
If(Location = remote) Outsider	If(Location = local) Insider

Table 25: Showing how threats are retrieved when the type of actor is human

Misuse Cases

This approach described [23,13] is an extension of use-case diagrams. A use case generally describes behavior that the system/entity owner wants the system to perform while Misuse cases apply the concept or behavior that the system’s owner does not want to occur. Use case diagram are driven by goals of the system misuse are driven by threats to the system. Figure 16 is showing the Misuse Case for Advanced Health Care System.

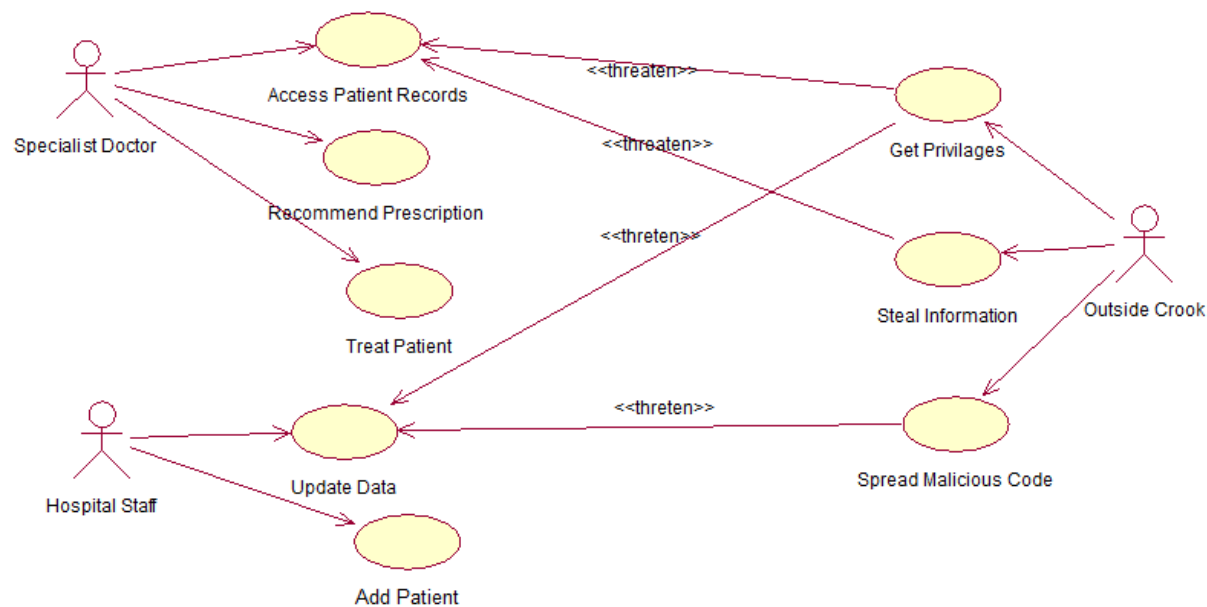


Figure 16: Misuse Case for Advanced Health Care System

Security Use Cases

This approach by Firesmith [8] says that misuse cases are highly effective ways of analyzing security threats but are inappropriate for the analysis and specification of security requirements, Because the success criteria for a misuse case is a successful attack against an application while the security use cases specify requirements that the application shall successfully protect itself from its relevant security threats. Figure 17 showing the concept of Security Use Cases with the example of Advanced Health Care System.

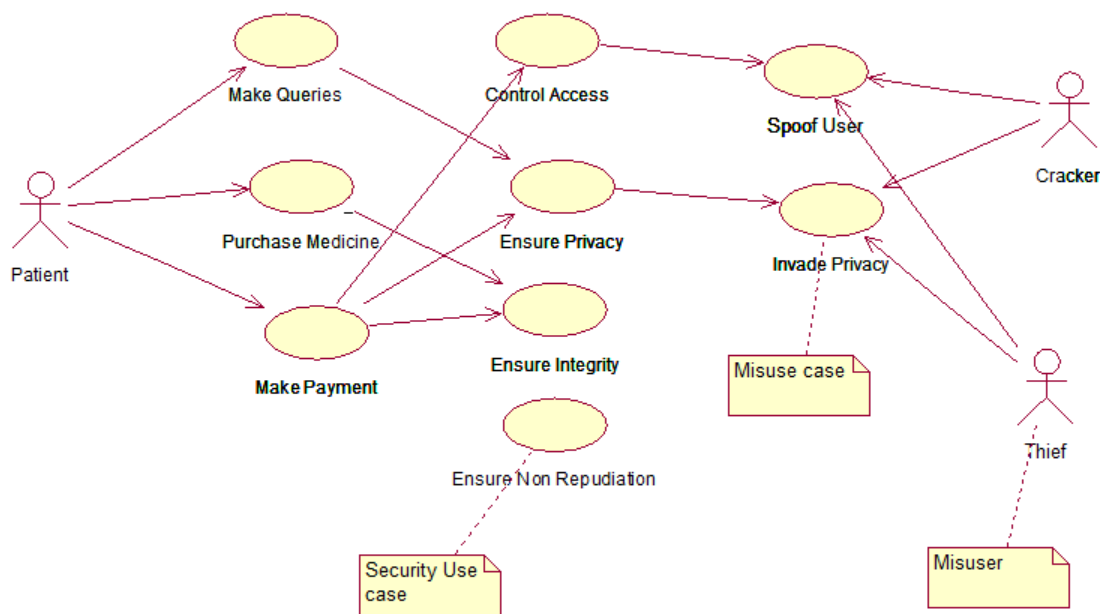


Figure 17 : Security Use Case for Advanced Health Care System

Attack Trees

Attacks trees [6] are a way to represent the attacks using the most widely used data structure Trees. Attack trees provide a formal, methodical way of describing the security of systems, based on varying attacks. Basically, you represent attacks against a system in a tree structure, with the goal as the root node and different ways of achieving that goal as leaf nodes. In this method the attack is represented with the attacker goal as the root node and the different ways of achieving that goal as leaf nodes. Satisfying a tree node represents either satisfying all leaves (AND) or satisfying a single leaf (OR). The value of attack tree analysis is derived from the attributes associated with each of the nodes. Figure 18; illustrate the simple attack tree for Advanced health care system.

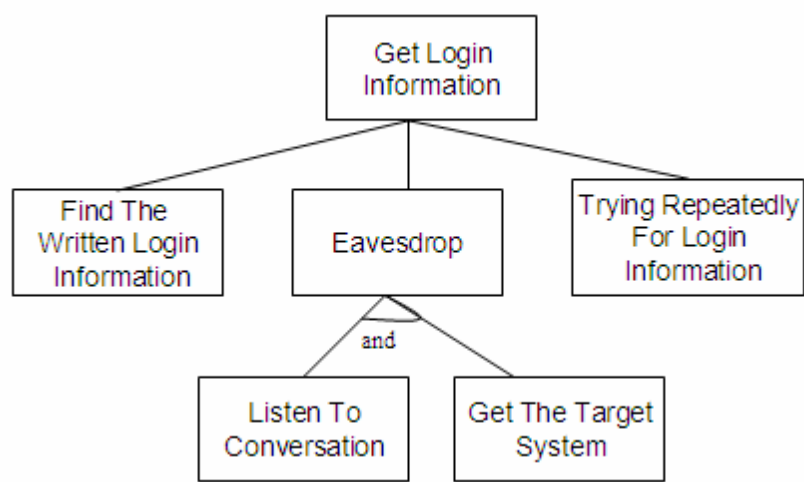


Figure 18: Attack Tree for Advanced Health Care System

In any real attack tree, nodes will have many different values corresponding to many different variables, both Boolean and continuous. Different node values can be combined to learn even more about a system's vulnerabilities. For instance, determines the cheapest attack requiring no special equipment. You can also find the cheapest low-risk attack, most likely nonintrusive attack, best low-skill attack, and cheapest attack with the highest probability of success, most likely legal attack, and so on. Every time you query the attack tree about a certain characteristic

of attack, you learn more about the system's security. So simple attack trees are extended to incorporate various other constraint for analysis.

To make this work, you must marry attack trees with knowledge about attackers. Different attackers have different levels of skill, access, risk aversion, money, and so on. If you're worried about organized crime, you have to worry about expensive attacks and attackers who are willing to go to jail. If you are worried about terrorists, you also have to worry about attackers who are willing to die to achieve their goal. If you're worried about bored graduate students studying the security of your system, you usually don't have to worry about illegal attacks such as bribery and blackmail. The characteristics of your attacker determine which parts of the attack tree you have to worry about.

ViewPoint Oriented Security Requirement Elicitation (VOSREP)

Here we would be describing the View point oriented method of eliciting security requirements given by Dr. Daya Gupta [6].

The VOSREP process defined is well embedded in VORD process making security engineering a unified approach with requirement engineering. Hence we can deal with security requirements as we deal with other functional and non –functional requirements.

In the VOSREP Process we give the techniques to elicit, analyze and manage security requirements.

The process VOSREP is based on following observation :

- Implementation of Security mechanisms effectively mitigate threats which can be considered as special kind of risk. Hence they can be assessed and analyzed using techniques from Risk assessment and risk analysis [35].
- In this VOSREP process Security requirements are driven from functionalities and data which are accessed by user of the system which may be internal or external to the system.

- Non functional requirements to some extent avoid security threats or cover security requirements.
- Security requirements are related to each other. For ex. - authorization requirements require existence of both identification and authentication requirements.

The different activities in the VOSREP process are as follows: -

3.3.1 Security Requirement Discovery and Definition

This is the first activity of the VOSREP process .The different steps in this activity are:

- I. Identify various stakeholders (actors) of the system using view-point analysis (Figure 2).
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 Engineering personals who develop software and people who regulate application domain. Our interest is in direct actor.

Example - for a 'Advanced Medical Care System' the direct actors/stakeholders can be Victim/Patient , Paramedics , Specialist Doctor, **Hospital Staff (Nurse / Receptionist)** etc. while the indirect actors can be System Administrators ,Maintenance Manager & Others

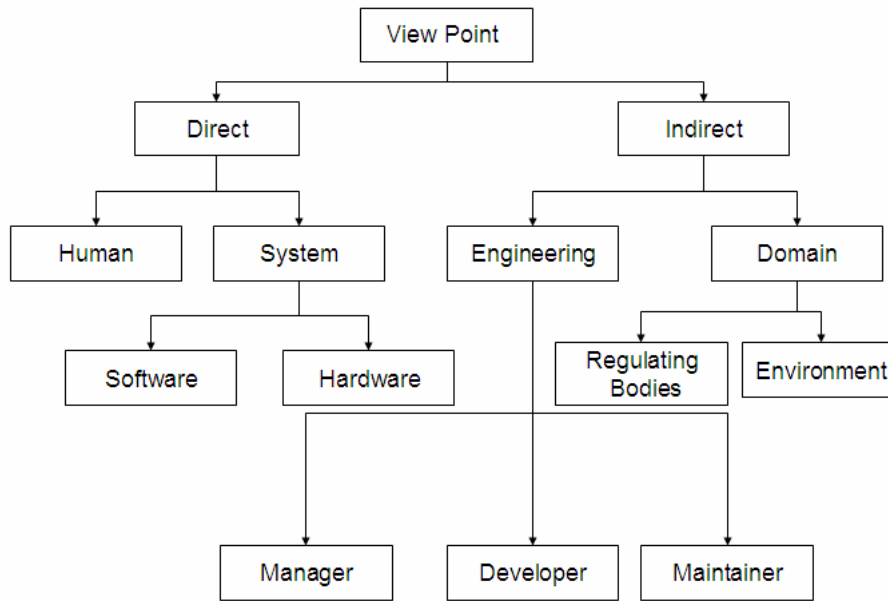


Figure 3 - Different types of stake holders as according to view point

- II. Identify the functionalities of each actor conceptualized in step i. Also determine associated non - functional requirements.

For ex- The functionality of **Paramedic** includes Assign Doctor, Inform Doctors, and Initial First Aid. Functionalities of **Specialist Dr** includes Emergency Case , Surgery case , Record procedure , Treat Patient , Recommend Prescription , Access Patients Reports . Functionalities of **Nurse/ Receptionist includes** Add patients, Assign Ward, Assign ID, Access Employee Database

- III. Identify the threats associated with each of the functional requirements or data which is used by this functionality. As in common criteria [14] based approach we shall be using predefined repository of the threats. Actor Profiles will be maintained as in common criteria based approach used for eliciting security requirements. Our actors will be classified as mentioned in view point analysis. Also threats will be classified, based on functionality and the actor kind predefined threats can be retrieved from the repository according to the profile of the actor.

To identify true security requirements first of all we will identify threats. To identify threats we will make a repository of threats. For each stakeholder we will make a stakeholders profile [24] that helps in the automatic generation of threats from the repository made by us. The profile of the stakeholder will be based on seven fields.

- Actor Name – Ex- Specialist Doctor , Paramedics
- Use Case – Ex- Add Patients, Access Patient Reports
- Type – Ex – Direct, Indirect etc.
- Location – Local Or Remote
- Private Exchange – Yes or No
- Secret Exchange – Yes or No
- Association – read, write, ask, answer, retrieve, store, send, display, update etc.

Example –

Stakeholder	Specialist Doctor
Functionality	Access Patients Reports
Type	Direct
Location	Remote
Private Exchange	True
Secret Exchange	True
Association	Write, Read, Update

Table 1 - Specialist Doctor Profile for Access Patient Reports

In general we can say that to identify threats we have correlated the Common Criteria (CC) with use case diagrams for the generation of threats based on stakeholders profile described above.

The repository that we have defined will be limited to the following category of threats [30, 24] shown in table 1.

	Threat Name	Description
1.	T.Change_Data	Information may be changed (insertions, replacements, modifications, deletions) while it being stored or processed.
2.	T.Data_Theft	Business process data may be stolen
3.	T.Deny_Service	Application and network services may not be available for use.
4.	T.Disclose_Data	Information may be disclosed in to unauthorized users while being stored or processed
5.	T.Impersonate	Someone may obtain unauthorized access by impersonating an authorized user.
6.	T.Insider	An authorized user may gain unauthorized access.
7.	T.Outsider	An individual who is not an authorized user of the system may gain access to the TOE.
8.	T.Privacy_Violated	Unauthorized access to privacy data of system users may occur without detection.
9.	T.Repudiate_Receive	An entity may deny that it has received business or commitment data.
10.	T.Repudiate_Send	An entity may deny that it has send business or commitment data.
11.	T.Spoofing	An entity may cheat for money.
12.	T.Social_Engineer	Tricking someone into giving you his or her password for a system than to spend the effort to hack in.

Table 2 – Threats Category and description

Once the threats have been identified we can define security requirements to mitigate these threats. The threat that have been identified in step (iii) above maps to security objectives which are mapped to security requirements and this is how true security requirements are identified. It is shown in Table 3.

Viewpoints	Services	Non-functional Requirements	Threats	Security Requirements
Specialist Doctor	1. Treat Patient 2. Surgery Case 3. Emergency Case 4. Access Patients Reports 5. Recommend Prescription 6. Record Procedure	1. Reliability with minimum system response time in giving assistance to specialist Doctor 2. Accuracy of information	T.Impersonate T.Data_Theft T.Disclose_Data T.Privacy_Violated T.Change_Data T.Repudiate_Receive	1. Authorization Requirement. 2. Privacy Requirements. 3.. Nonrepudiation Requirements
Nurse / Receptionist	1. Add Patient 2. Access Employee Database 3. Assign wards 4. Assign Patient Id	1. Correctness of Information. 2. Minimize response time in accessing patients records..	T.Change_Data T.Privacy_Violated T.Social_Engineer T.Outsider	1. Integrity Requirements. 2. Authentication Requirements 3. Identification Requirements

Table 3 : Example of “Advanced Medical Care System” using VOSREP

3.4 Security Requirement Analysis & Prioritization

3.4.1 Analyzing Security Requirements

The various steps in this analyzing the security requirements are as follows:-

- **Checking For completeness**

In this step we will make a check list to check that the security requirements that have been elicited have mitigated all the threats to the functionality of the system.

It means that we just check that all the threat have been taken in to consideration or not, because if any of the threat has been left over it can cause the system to fail or can be attacked in later stages. We simply create a table containing a list of threats to the stakeholders and put yes/no against each threat if the threat has been checked or not respectively.

- **Checking for Consistency**

In this step we resolve the contradictions that may exist in the security requirements elicited from different viewpoints. This is very essential to ensure that the final specifications of security requirements are consistent.

In this step we also check that the security requirement for realism i.e. security requirements can be implemented in some or other way in the budget of the project.

- **Group Related Requirements**

This step consists of identifying the security requirements that can be grouped together. We group them according to if one of the security requirements are implemented the others in the group are implemented automatically.

3.4.2 Security Requirement Prioritization

After the security requirements have been analyzed for completeness, consistency and are grouped together then we prioritize them by following simple steps. We will use the CRAMM process of risk analysis to prioritize security requirements. The steps for prioritizing security requirements are as follows:-

- Prioritize each threat to the asset using any the risk measuring techniques such as CRAMM [20], OCTAVE [18], or CORAS [19].
- Once the threats have been prioritized we have to back track threats to security requirements so that we can prioritize security requirements. The security requirement corresponding to highest priority threat will be the highest priority security requirement.

We use the CRAMM process for risk measuring, with the following steps –

3.4.2.1 Evaluation of Threats

We will evaluate threats based on the calculated value of risk. In this task we have to follow the following steps that are as follows and as shown in Figure 4.

- i) Threat Assembling
- ii) Threat Rating
- iii) Assigning value to corresponding vulnerability
- iv) Identify the concerned affected Asset and give them a value.
- v) Estimate the value of Risk

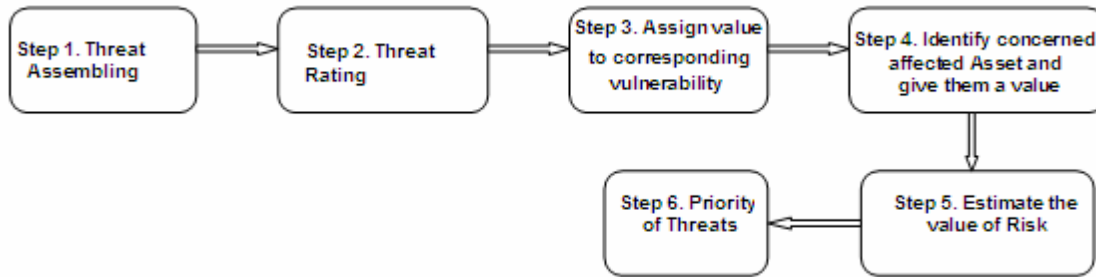


Figure 4 : Process for Threat Priority

Now all the above steps are defined below and are explained with an example of Railway Reservation system. We have chosen CRAMM method for identification of risk as it offers structured and fast approach to risk analysis over other methods [36].

Step1. Threat Assembling – Assemble threats which are a source of each security requirements. As in common criteria based approach [3, 24] we shall be developing the repository of the threats. Actor Profiles will be maintained. Predefined threats can be retrieved from the repository according to the profile of the actor that is defined in section 3.2.

Step2. Threat Rating – It is the value which defines the probability of occurrence of particular threat in a project. We have already defined the process of threat assembling, so whatever threats have been identified we have to assign them a value according to CRAMM [29]. These values can be very low (.1), low (.34), medium (1), high (3.33) and very high (10) depending on their occurrence.

Step3. Assigning value to corresponding vulnerability – Vulnerability is defined as the weakness in the system that makes an attack more likely to succeed. Value of vulnerability defined by CRAMM [29] will be taken that are low (.1), medium (.5) and high (1). The value can be assigned according to following Table 4.

Vulnerability Rating	Guide
Low (0.1)	An incident is expected to occur, there would be no more than a 33 % chance of the worst case scenario (assessed during asset evaluation) being realized.
Medium (0.5)	If an incident were to occur, there would be 33% to 66 % chances of the worst case scenario (assessed during asset valuation) being realized.
High (1.0)	If an incident were to occur , there would be higher than 66 % chances of the worst case scenario (assessed during asset valuation) being realized.

Table 4: Measure of Vulnerability

Step4. Identify the concerned affected Asset and give them a value – An asset can be anything that has a value to an organization (e.g. IT systems, information, staff, reputation, goodwill). Different asset in the project is identified and their value is measured by weighing the impact of it when threat will occur.

Step5. Estimate the risk level – Risk is defined as the probability that a threat agent (cause) will exploit system vulnerability (weakness) and thereby create an effect detrimental to the system, or in short risk is an unwanted event that has negative consequences on the system. We can say that Risk is defined in one line as (1) –

$$\text{Risk} = \text{value based on measure of (Threat, Vulnerability, Asset)} \quad (1)$$

After we have rated the threats, assigned vulnerability value and asset value we will use the 3 dimensional lookup table given by the CRAMM [29] shown in Table 5, where the strength of the threat, the level of the vulnerability and the value of the asset are input parameters, gives the final value of risk in the range 1 through 7.

Threat Rating →	.1	.1	.1	.34	.34	.34	1	1	1	3.33	3.33	3.33	10	10	10
Vulnerability →	.1	.5	1	.1	.5	1	.1	.5	1	.1	.5	1	.1	.5	1
Asset Value ↓															
1	1	1	1	1	1	1	1	1	2	1	2	2	2	2	3
2	1	1	2	1	2	2	2	3	3	2	3	3	3	3	4
3	1	2	2	2	2	3	2	3	3	3	3	4	3	4	4
4	2	2	3	2	3	3	3	3	4	3	4	4	4	4	5
5	2	3	3	3	3	4	3	4	4	4	4	5	4	5	5
6	3	3	4	3	4	4	4	4	5	4	5	5	5	5	6
7	3	4	4	4	4	5	4	5	5	5	5	6	5	6	6
8	4	4	5	4	5	5	5	5	6	5	6	6	6	6	7
9	4	5	5	5	5	6	5	6	6	6	6	7	7	7	7
10	5	5	6	5	6	6	6	6	6	6	7	7	7	7	7

Table 5– Three – Dimension lookup table to measure the level of risk

3.4.2.2 Prioritizing Security Requirements

Prioritization of security requirement incurs the following step shown in Figure 5.



Figure 5: Security Requirement Prioritization

Step1. Identify The Security Requirements – Identify the various security requirements corresponding to threats [14, 57] so that we can give them priority to mitigate the threats.

Step2. Backtrack to find out priority of security requirement – Although till now all the process of analysis and prioritization of requirements are the steps of CRAMM, this activity is the real part of our process.

When we have identified the measures of risk to all the threats and prioritize them based on value of risk. The more the value higher is the priority. We have to consider various cases while prioritization of security threat.

Case1: Simple Priority – We have only one asset corresponding to a threat then its risk measure will be calculated based on equation (1). Whatever value comes of risk measure is the priority for that threat.

Case2: Complex Priority - We have two assets corresponding to a threat. Then we have two values for this threat corresponding to two assets whose risk values are also comes to be two. So priority for this threat will come by adding the two values.

Finally we have arrived to our real task that is the calculation of priority of Security Requirement. We will calculate the priority of Security Requirement just from the value of threat priority. The following case may arise –

Case1 – It is the sum of the value of threat priority, if corresponding to one security requirement there are more than one threat.

Case2 – If there is only one threat corresponding to security requirement then what so ever is the value of threat priority that will be assigned to the security requirement.

As we have the priorities of Security Requirement now the developer will check what are the Security Requirement corresponding to the various actors that are listed in Table 3 and correspondingly deal with them.

3.4.3 Managing Security Requirements

As we manage functional, non- functional and other requirements we have to manage security requirements too. If we do not manage the security requirements with other activities of the system under development we led to a system that will not be efficient and cost effective.

To manage security requirements we have to keep trace of each security requirements and its associated attributes such as requirement identity, view point identity, functional requirement, nonfunctional requirements, threats, design constraint, other security requirement, design constraints etc. This information is modeled in figure 6.

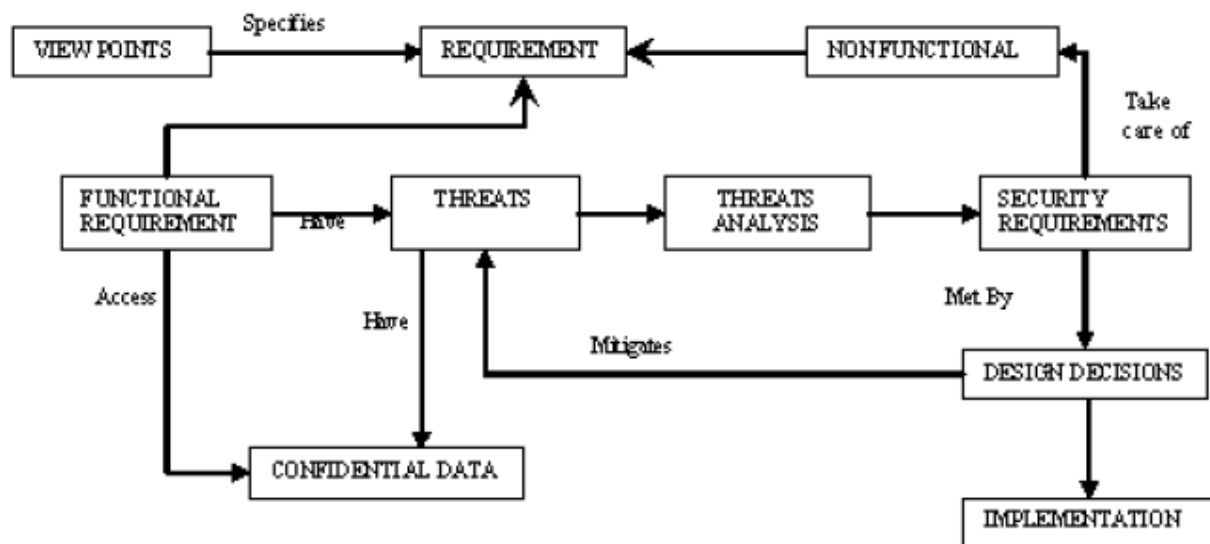


Figure 6 – Model for Managing Security Requirements

There are three types of traceability information that must be maintained for the management of security requirements.

- Source Traceability.
- Security Requirement Traceability.
- Design Traceability.

3.4.3.1 Source Traceability

This information for traceability refers to the information of threats on the functionality of the system that misuser can create for the system from where we have derived our security requirements as modeled in figure above the cause of security requirements is the threats to the functionality of the system.

3.4.3.2 Security Requirement Traceability

This traceability information refers to information of

- Functional requirements that are the root cause of the security requirements.
- Interdependent Functional Requirements.
- Interdependent security requirements.

If the main security requirement changes the dependent security requirements also tends to change.

3.4.3.3 Design Traceability

This information about traceability links to the design modules where they are going to be implemented. The information is kept so that the corresponding design decisions that have been taken are actually be implemented.

Traceability information for security requirements can be managed using traceability metrics which relate security requirements to stakeholders (in our case as according to view point), each other and design modules.

3.5 Summary

In this section we discussed the various techniques for eliciting security requirements including the VOSREP (Viewpoint Oriented Security Requirement Elicitation Process) technique, CRAMM technique for prioritizing the identified security requirements and management of security requirements. This section provides the background knowledge to understand the later section which deals with analyzing the security requirement for taking design decision that would help mitigate these identified security threats.

CHAPTER 4 Security Engineering & Cryptography

Introduction –

From the security engineering perspective, cryptography forms an important factor in the design considerations of building a secure system [46]. Threats or the security requirements that are identified during the initial stage of the security requirement engineering need to convert into design decision that would help mitigate the identified threats within the system.

Information assurance , the body of knowledge, policies ,processes , practices & tools that provide reasonable assurance that one's Information and communications are used only as intended and only by authorized parties , has become a complex discipline. Certain Architectural & Security requirements play an important role in evaluating & finding the best security protocol, based upon the various security attributes of the system.

The actual application of cryptography protocols, and the practical implementation of Cryptosystem primitives in the real world constitute a complex and undoubtedly interdisciplinary research field, involving mathematics and computer science as well as electrical engineering. Investigation of protocols robustness to software and hardware implementation constitutes a complex subject of interest in itself. In general, the role of the implementation efficiency requirement (i.e. the possibility to achieve suitable level of execution time, resource required, power consumed, costs along with flexibility and reusability of design solutions) is rather clear.

What is somewhat implicit in the overall picture concerns the security aspect and is related to the fact that, surprisingly, implementation generates a significant, perhaps crucial portion of the real security risk. **In fact, it is widely recognized that most of the threats inherent in real-world security infrastructure stems from how we perform cryptographic operations on secret data rather than from the intrinsic mathematical strength of the cryptosystem. As a consequence , architectural and implementation details , which normally take on a marginal**

role and seemingly just affect performance , are of crucial importance , as they could open the door to many real-world attacks in a number of nontrivial and often unforeseen ways .In this chapter , we explore how these aspects have a crucial role in steering design choices.

4.1 Cryptographic Engineering -

The purpose of cryptography is to render information unintelligible to all but the intended receiver. The sender enciphers a message into unintelligible form, and the receiver deciphers it into intelligible form. The word “ cryptology “ is derived from the Greek **Kryptos** (hidden) and **logos** (word)

Cryptology: The scientific study of cryptography and cryptanalysis

Cryptography: The enciphering and deciphering of message into secret codes by means of various transformation of the plain text.

Cryptanalysis: The process of deriving the plaintext from the cipher text (breaking a code) without being in possession of the key or the system (code breaking).

Generally speaking, the strength of a public-key cryptosystem is directly related to the type of the one-way function it uses and the length of the cryptographic keys. With the computing power and the theoretic knowledge available today, we find that inverting a one-way function—the scalar multiplication for the case of EC cryptography—is a practically intractable problem. It is interesting to note that, for hard problems commonly used in cryptography, such as the RSA factoring problem [45] and the elliptic curve discrete logarithm problem [44], this is just conjectured to be true, as no mathematical proof exists about the intrinsic complexity of such problems.

On the other hand, many specific algorithms exist to solve cryptographic hard problems used for public-key cryptosystems, and often their levels of computational complexity are pragmatically taken as a measure of the cryptosystem strength, even though it is not excluded that more efficient algorithms may exist. It has also been found that many feasible methods exist to attack particular instances of the hard problems (for example, the discrete logarithm

problem over supersingular elliptic curves [44]). The choice of the underlying mathematics, domain parameters, and key sizes is strongly influenced by such results when new cryptographic schemes and standards are defined. As long as all these choices are made appropriately, and no breakthroughs in number theory or computing technologies are achieved, the *key size* is the cryptosystem parameter that can be used for scaling the robustness of the cryptosystem over time or according to actual application needs. **In fact, the key size [43] usually affects the complexity of the underlying cryptographic hard problem, and thus determines the overall mathematical strength of the cryptosystem.** In principle, we could obtain a cryptosystem as secure as we want by just increasing the key length at will.

However, in practical applications, there are some other considerations to do. First, in addition to being mathematically strong, the cryptosystem should be practically feasible. In fact, implementation efficiency depends on key sizes, as the cryptosystem security, although they follow different laws in general. A mathematically strong cryptosystem whose implementation requires prohibitive computation resources is of no practical utility.

In a wider sense, implementation efficiency could be seen as the property that allows a particular design solution, either software or hardware, to be used, and reused, in a scalable, modular, and flexible way. Most of these requirements are inherent in cryptographic algorithms and applications. In fact, the criteria of reusability and scalability are of fundamental importance for the design of cryptographic blocks, since the operand sizes, usually much larger than in normal applications, may significantly change depending on the required level of security or the specific cryptosystem.

But there is another dimension in cryptographic engineering, distinct from mere implementation efficiency, that only recently has been fully recognized. This dimension has essentially to do with security and stems from the fact that, unfortunately, we cannot think of the *implementation* of a cryptographic primitive holding secret data (e.g., an integrated circuit card) as a perfect black box. Implementations do leak sensitive information, and such leakage might be far more significant than a possible unforeseen mathematical flaw in a cryptographic

one-way function. In other words, cloning a smart card may be not as difficult as breaking the internal public-key algorithm, no matter how mathematically strong it is.

Both implementation efficiency and implementation security are essential aspects of cryptographic engineering, which we could indeed define as the science of translating cryptographic algorithms into feasible and secure design solutions. **In the rest of the thesis , we will show how cryptographic engineering requirement influence design choices at the various layers of security engineering .**

4.2 Cryptographic Services -

Different types of security services provided by cryptography are - (1) privacy or confidentiality (2) data integrity (3) authentication and (4) non-repudiation.

1. Confidentiality is a service used to keep the content of information from all but those authorized to have it. *Secrecy* is a term synonymous with confidentiality and privacy. There are numerous approaches to providing confidentiality, ranging from physical protection to mathematical algorithms which render data unintelligible.

2. Data integrity is a service which addresses the unauthorized alteration of data. To assure data integrity, one must have the ability to detect data manipulation by unauthorized parties. Data manipulation includes such things as insertion, deletion, and substitution.

3. Authentication is a service related to identification. This function applies to both entities and information itself. Two parties entering into a communication should identify each other. Information delivered over a channel should be authenticated as to origin, date of origin, data content, time sent, etc. For these reasons this aspect of cryptography is usually subdivided into two major classes: *entity authentication* and *data origin authentication*. Data origin authentication implicitly provides data integrity (for if a message is modified, the source has changed).

4. Non-repudiation is a service which prevents an entity from denying previous commitments or actions. When disputes arise due to an entity denying that certain actions were taken, a

means to resolve the situation is necessary. For example, one entity may authorize the purchase of property by another entity and later deny such authorization was granted. A procedure involving a trusted third party is needed to resolve the dispute.

A fundamental goal of cryptography is to adequately address these four areas in both theory and practice. Cryptography is about the prevention and detection of cheating and other malicious activities.

4.3 Security Attacks –

Cryptographic attacks [42,53] are designed to subvert the security of cryptographic algorithms , and they are used to attempt to decrypt data without prior access to a key .

4.3.1 Cryptographic Attack Methods –

1) Known Plain text and Cipher text only attacks –

A known plaintext attack is an attack where a cryptanalyst has access to a plaintext and the corresponding cipher text and seeks to discover a correlation between the two.

A cipher text-only attack is an attack where a cryptanalyst has access to a cipher text but does not have access to corresponding plaintext. With simple ciphers, such as the Caesar Cipher, frequency analysis can be used to break the cipher.

2) Chosen Plaintext and Chosen Cipher Text Attacks -

A chosen plaintext attack is an attack where a cryptanalyst can encrypt a plaintext of his choosing and study the resulting cipher text. This is most common against asymmetric cryptography, where a cryptanalyst has access to a public key.

A chosen cipher text attack is an attack where a cryptanalyst chooses a ciphertext and attempts to find a matching plaintext.

3) *Side Channel Attacks* -

Side channel attacks leverage additional information based on the physical implementation of a cryptographic algorithm, including the hardware used to encrypt or decrypt data.

Many practical side channel attacks have been discovered. One example is the network-based attack versus OpenSSL.

4) *Brute Force Attacks*-

A brute force attack systematically attempts every possible key. It is most often used in a known plaintext or ciphertext-only attack. Here is an example of a brute force attack on a 4-bit key:

0000	0001	0010	0011	0100	0101	0110	0111
1000	1001	1010	1011	1100	1101	1110	1111

Given a finite key length and sufficient time, a brute force attack is always successful. Encryption algorithms can become susceptible to brute force attacks over time as CPU speeds increase. Single DES encryption has an effective key length of 56-bits, and any key can be cracked within days using specialized hardware, such as the Electronic Frontier Foundation's Deep Crack. [31] Triple DES (168-bit key) was approved due to DES's weakness to brute force attacks, followed by the Advanced Encryption Standard (AES) in 2001. If a machine could crack one DES key per second, it would take 149 thousand-billion (149 trillion) years to crack a 128-bit AES key[32].

5) *Meet-in-the-Middle Attack* -

Meet-in-the-middle attacks can be used against cryptographic algorithms that use multiple keys for encryption. An example of a successful meet-in-the-middle attack is the attack versus Double DES. The meet-in-the-middle attack is a known plaintext attack; the cryptanalyst has access to both the plaintext and resulting ciphertext.

6) *Linear Cryptanalysis and Differential Cryptanalysis*

Differential cryptanalysis and linear cryptanalysis are related attacks used primarily against iterative symmetric key block ciphers. An iterative cipher (also called a product cipher) conducts multiple rounds of encryption using a subkey for each round. Examples include the Feistel Network used in DES and the State rounds used in AES. In both attacks, a cryptanalyst studies changes to the intermediate ciphertext between rounds of encryption. The attacks can be combined, which is called differential linear cryptanalysis.

- ***Linear Cryptanalysis***

Linear cryptanalysis is a known plaintext attack that requires access to large amounts of plaintext and ciphertext pairs encrypted with an unknown key. It focuses on statistical analysis against one round of decryption on large amounts of ciphertext.

The cryptanalyst decrypts each ciphertext using all possible subkeys for one round of encryption and studies the resulting intermediate ciphertext to seek the least random result. A subkey that produces the least random intermediate cipher for all ciphertexts becomes a candidate key (the most likely subkey).

- ***Differential Cryptanalysis***

Differential cryptanalysis is a chosen plaintext attack that seeks to discover a relationship between ciphertexts produced by two related plaintexts. It focuses on statistical analysis of two inputs and two outputs of a cryptographic algorithm.

A plaintext pair is created by applying a Boolean exclusive or (XOR) operation to a plaintext.

4.3.2 Implementation Specific Attacks –

These attacks exploit the implementation specific characteristics rather than mathematical properties of the cryptosystem attacked. The study of Implementation specific attacks is a fundamental part of cryptographic engineering and is essential for the design of real-world cryptographic devices.

Implementation attacks are usually divided into *invasive* and *noninvasive* attacks. With invasive attacks the cryptographic device is physically tampered with using some special equipment. A general methodology for hardware devices consists in unpackaging the chip and reverse engineering the silicon layout by means of optical microscopes and microprobing workstations. Invasive attacks require a specialized laboratory and time-consuming reverse engineering operations and they always destroy the packaging of the card. On the other hand, they require very little initial knowledge and usually work with a similar set of techniques on a wide range of products [33].

Noninvasive, or side-channel attacks are based on the general idea of measuring some side-channel information on tamper-resistant devices, such as power consumption or timing, and attempting to infer secret keys from such information. Noninvasive attacks may be particularly dangerous. In fact, although the attack requires physical access to the device, usually it is not evident so that it is unlikely that the validity of the compromised data will be revoked before they are abused. In addition, side-channel attacks can be mounted with relatively inexpensive equipment. On the other hand, most noninvasive attacks require some preliminary knowledge of hardware and software aspects of the attacked device.

Noninvasive attacks can be divided into the following categories.

- **Timing Analysis Attacks.** Timing attacks [34] are based on measuring the time that the unit under attack requires to perform specific operations. They are based on the fact that, in straightforward implementations, many steps with different execution times may or may not be executed depending on the handled data, including secret keys.

- **Power Consumption Attacks.** Power attacks [37] require the interpretation of power consumption measurements collected during cryptographic operations and are based either on *Simple Power Analysis (SPA)* or *Differential Power Analysis (DPA)*. SPA can be used to break cryptographic implementations in which the execution path depends on the data being processed, similar to timing attacks. DPA attacks use statistical analysis and error correction techniques to extract information correlated to secret keys.
- **Electromagnetic Attacks.** Electromagnetic attacks [39] are based on the same principle of power attacks applied to the analysis of electromagnetic radiation. They are classified into *Simple Electromagnetic Analysis (SEMA)* attacks and *Differential Electromagnetic Analysis (DEMA)* attacks.
- **Fault Analysis.** Fault Analysis techniques [38] consist in tampering with a device and making it perform some faulty computation that, when properly analyzed, may leak information about the secret parameters involved in the computation. Faults are induced by using heat, pressure, external electromagnetic fields, clock glitches, or power supply transients.

4.4 Analysis Of Strength Of Various Cryptography Protocols -

The purpose of this section is to provide an in-depth analysis of various security protocols and investigate their operational and performance feasibility in different environments. To provide a small representative sampling of well known cryptographic algorithms, six were originally selected: five symmetric or secret key (one-key) block ciphers and one asymmetric or public key (two-key) algorithm. Norman D.Jorstad [53] has proposed following metrics for evaluating and comparing cryptography algorithms.

4.4.1 Metrics for Characteristics of Cryptography Protocol

1. **Type** - *symmetric* (secret key or one-key) or *asymmetric* (public key or two-key). While strictly speaking this may not be a metric, the type of key that an algorithm uses would be of sufficient interest to users to be worth specifying. (Because there are short-

cut attacks that can be used on asymmetric algorithms, very long keys are required. With any meaningful key length, two-key algorithms are very slow when compared to one-key algorithms. This effectively limits their use to the management of keys for symmetric algorithms. It should be noted that some two-key algorithms can provide a covert channel for traffic while masquerading as signatures.)

Example – Symmetric key algorithms like AES, DES are 1000 times faster than Asymmetric algorithms and commonly used for confidentiality services by encrypting & decrypting data . Asymmetric key algorithms like RSA, Diffie-Hellman, ElGamal are slower and computation intensive hence used commonly for exchanging keys securely.

2. Functions - Message secrecy, message integrity, authentication, digital signatures, like the type of algorithm, may not be a metric but may be of interest to end users.

3. Key size - The Key Length Metric indirectly indicates the strength of the protocol. The security of a symmetric cryptosystem is a function of the length of the key. The longer the key, the more resistant the algorithm is to a successful brute force attack. For this reason, key length was chosen as the first parameter for specifying cryptographic algorithms. Key Length is an easy objective, numeric metric to adopt since key size is universally expressed as a number of bits. For example, the standard key length for the Data Encryption Standard (DES) is 56 bits.

4. Rounds - Rounds were considered but may not be an important metric because rounds, like word and block size, are not universal characteristics and may not have great value in specifying meaningful thresholds.

5. Complexity - (Algorithm complexity for encryption, decryption, and key setup.) These attributes for encryption, decryption and key setup probably could be specified as the number of operations such as bit operations, modular multiplications and modular exponentiations. The number of operations wouldn't change, only the speed of implementation

6. Attack. - Best known methods of attack such as brute force, factoring, linear and differential cryptanalysis (qualified with whether known or chosen plaintext is provided,) number of steps and time required for a successful attack.

(i) Attack Time - Attack Time is defined as the time required to perform the fastest known attack on a specified processor.

(ii) Attack Step - Attack Steps is defined as the number of steps required to perform the best known attack. The number of steps helps determine the time that might be required for a successful attack, using a particular processor, without having to actually run the attack on the algorithm, which may not be feasible.

7. Strength - An assessment of the strength of the algorithm, based on key length, algorithm complexity and the best methods of attack.

4.4.2 Application

4.4.2.1 DES (DEA) -

The Data Encryption Standard (DES), a.k.a. the Data Encryption Algorithm (DEA), was the first symmetric block cipher chosen because the DES is a long standing federal standard [40] and the DEA has been adopted by the American National Standard Institute (ANSI) as a standard and is incorporated in several international standards. DES has been extensively studied since it was first issued as a standard in 1977 and found to be mathematically sound. As required by the standard, DES has been reviewed every five years and was reaffirmed in 1983 and 1988. The third review was conducted in 1993 and DES has been reaffirmed (for the third time) until 1998 as Federal Information Processing Standard Publication (FIPS PUB) 46-2. The National Institute of Standards and Technology (NIST) believes that DES provides adequate security for its intended *unclassified information*¹³ applications. The DES algorithm can be used in any one of the four operating modes defined in FIPS 81.

In typical applications, DES is used to encrypt relatively large volumes of data for interchange, using a key which is exchanged securely (often done using a public key cryptographic protocol). Since DES operates on blocks of data which are only 8 bytes in length and the size of the data that needs to be encrypted or decrypted is often quite large, the speed of encryption/decryption in a DES implementation is an important factor.

Attributes Analysis for DES –

Complexity	The algorithm specified in the DES is very complex
Encryption	It encrypts data in 64 bit blocks, using a 56-bit secret key.
Keys	There are 2^{56} or about 7.2×10^{16} possible keys for DES. There are four “weak” keys that, if selected, may decrease the security of DES by a factor of two. (When keys are randomly generated, an adversary never knows if a “weak” key is used, hence no keys are weak for a well designed system.)
Attacks	The best attacks on DES that are known are the brute force attack and differential and linear cryptanalysis. (Differential and linear cryptanalysis both are computationally complex.)
Application	Confidentiality Services Not secure. Recommended AES with 128bit key size instead of DES.

4.4.2.2 3DES (Triple DES) -

Triple DES (3DES), a.k.a. Encrypt-Decrypt-Encrypt16 (EDE) and the Triple Data Encryption Algorithm (TDEA), is the name now most often given one popular form of multiple DES applications. Most 3DES implementations use two keys; however, 3DES can use two or three keys. Since DES is (mathematically) not a group [41], the resultant 3DES (using two keys) ciphertext is much harder to break using the exhaustive search method; 2^{112} , instead of 2^{56} attempts are required. There is not complete agreement among authorities that the effective key length of 3DES is really 112. To meet the requirements of the financial community for stronger cryptography, while preserving their investment in DES, ANSI Working Group X9.F.1 is developing American National Standard X9.52 - 19XX, *Triple Data Encryption Algorithm and Modes of Operation*. The X9.52 ANSI standard is expected to include modes that will allow two or three different keys, which would produce an effective key length of 112 or 168 bits, respectively.

Attributes Analysis for 3DES –

Complexity	Same complexity as DES with multiple round of operations
Encryption	It encrypts data in 64 bit blocks, using a 112-bit or 168-bit secret key.
Keys	There are 2^{112} possible keys for Triple DES.
Attacks	The best attacks on DES is differential cryptanalysis attack . The attack complexity increases by a factor of 4 for Triple DES or by a factor of $2^{(n-1)}$ for n number of DES encryption .(Differential and linear cryptanalysis both are computationally complex.)
Cryptography Services	Confidentiality Services

4.4.2.3 RC5

RC5 was chosen because it is a new fast, symmetric block cipher and the dimensions for the metrics proposed in this pilot were readily available in Dr. Rivest's article in the January 1995 issue of *Dr. Dobbs's Journal*. The RC5 is suitable for both hardware or software implementations. A novel feature of RC5 is the heavy use of data-dependent rotations. RC5 has a variable-length secret key, providing flexibility in its security level. It is a parameterized algorithm, and a particular RC5 algorithm is designated RC5-w/r/b, where w is the word size in bits, r is the number of rounds, and b is the number of bits in the secret key. RC5 provides for the use of up to a 255 bit key.

Since RC5 uses variable length keys and number of rounds, statements that are made about its strength must be made in relation to specified key lengths and rounds; and, Algorithm Strength ratings must be qualified with a specification of these variables.

Attributes Analysis for RC5 –

Complexity	High
Keys	Variable length key , upto 255-bit key
Application	Suitable for H/w & S/w implementation
Attack	Differential Attack
Cryptography Services	Confidentiality Services

4.4.2.4 RSA

The Rivest-Shamir-Adleman (RSA) asymmetric public key cipher is named for its creators: R.L. Rivest, A. Shamir and L.M. Adleman, who were all members of the Massachusetts Institute of Technology (MIT) Laboratory for Computer Science when they developed the public key implementation of the Diffie-Hellman concept. RSA was chosen because it is popular and has been extensively analyzed. RSA is a widely advertised commercial public key algorithm used in business and personal communications. The RSA variable key size may be anywhere from 2 to 2,048 bits in current implementations. The security of these algorithms depends on the key size that the user or programmer chooses.

Attributes Analysis for RSA –

Complexity	Medium
Keys	768 Bit RSA – Personal Use 1024 Bit RSA – Corporate Use 2048 Bit RSA – For Protecting Extremely Valuable Data
Attacks	Exhaustive Key Search Attack Integer Factorization (Including Quadratic Sieve , the Generalized Number Field Sieve, and the Lattice Sieve)
Strength	Based upon Integer Factorization Problem
Cryptography Services	RSA – Confidentiality Services (Encryption / Decryption) RSA Digital Signatures – Non Repudiation / Authentication

4.4.2.5 ECC (Elliptic Curve Cryptography)

An elliptic curve is defined by an equation in two variables, with coefficients. For cryptography, the variables and coefficients are restricted to elements in a finite field, which results in the definition of a finite abelian group. Elliptic curve cryptography makes use of elliptic curves in which the variables and coefficients are all restricted to elements of a finite field.

Two families of elliptic curves are used in cryptographic applications: prime curves over \mathbb{Z}_p and binary curves over $\text{GF}(2^m)$.

For a **prime curve** over \mathbb{Z}_p , we use a cubic equation in which the variables and coefficients all take on values in the set of integers from 0 through $p-1$ and in which calculations are performed modulo p .

For a **binary curve** defined over $\text{GF}(2^m)$, the variables and coefficients all take on values in $\text{GF}(2^n)$ and in calculations are performed over $\text{GF}(2^n)$.

Fernandes [58] points out that prime curves are best for software applications, because the extended bit-fiddling operations needed by binary curves are not required; and that binary curves are best for hardware applications, where it takes remarkably few logic gates to create a powerful, fast cryptosystem. Elliptic curve arithmetic can be used to develop a variety of elliptic curve cryptography (ECC) schemes, including key exchange, encryption, and digital signature.

Attributes Analysis for ECC –

Keys	112 Bit ECC – Personal Use 160 Bit ECC – Corporate Use 224 Bit ECC – For Protecting Extremely Valuable Data
Attacks	Power Analysis Attack Fault Attacks when running on Smart Cards [59]
Strength	Elliptic curve discrete logarithmic problem
Cryptography Service	ECC – Confidentiality Services ECC with DSA (ECDSA) – Non-Repudiation & Authentication ECC DH(Diffie Hellman)- Key Exchange / Authentication

4.4.2.6 HECC (Hyper Elliptic Curve Cryptography)

Neal Koblitz [60] proposed the hyperelliptic curve cryptosystem (HECC) in 1989 , which is based on the discrete logarithm problem on the jacobian of hyperelliptic curves over finite field . At the same level of security , the underlying field of HECC is smaller than that of ECC , and almost all the standard discrete logarithm based protocols such as the digital signature algorithm (DSA) and ElGamal can be planted to HECC. Many theoretical results on elliptic curve are known , however , the known result on hyperelliptic curve are still not enough for the construction of efficient cryptosystem.

Some of the results by Jan Pelzl [62] have shown that HECC with binary field are more suitable for embedded system due to low field order.

In case of HECC we need 40-bits to 80-bits long operand to compute the group operations for these curves. In the case of ECC we have work with operands length of approximately 160 bits. Whereas in the case of RSA , the operand will be approximately 1024 bits in order to achieve the same security[61]. Following data were obtained from our implementation of HECC genus 2 and genus 3 curves over binary and prime fields.

Attributes Analysis for HECC –

Keys	HECC G(2) 80 Bit / HECC G(3) 54 Bit– Corporate / Critical Use For personal Use , Lower operand size can be chosen depending upon the requirement.
Strength	Elliptic curve discrete logarithmic problem
Cryptography Service	HECC P / HECC B – Confidentiality Services HECC B – Suitable for Embedded Systems HECC with DSA (ECDSA) – Non-Repudiation & Authentication

4.5 Comparison

Different cryptography services like confidentiality/privacy , Integrity , Authentication & Non-Repudiation are provided by different types of algorithm . Comparative studies in the previous sections have shown that symmetric key algorithms are best suited for encryption/ decryption and Asymmetric algorithms for key distribution and authenticity.

Table 24 compares various algorithms by showing comparable key sizes in terms of computational effort for cryptanalysis. As can be seen, a considerably smaller key size can be used for ECC compared to RSA. Furthermore, for equal key lengths, the computational effort required for ECC and RSA is comparable [58]. Thus, there is a computational advantage to using ECC with a shorter key length than a comparably secure RSA.

Symmetric Scheme (key size in bits)	ECC-Based Scheme (size of n in bits)	RSA/DSA (modulus size in bits)
56	112	512
80	160	1024
112	224	2048
128	256	3072
92	384	7680
256	512	15360

Table 24: Comparable Key Sizes in Terms of Computational Effort for Cryptanalysis , Source: Certicom

Table 25 gives a analysis of different cryptography algorithms’ and the services provided by them . These Cryptography services can then be mapped to appropriate security requirement and the best available algorithm be chosen.

Cryptography Services	Typical Algorithms	Functions	Limitations
Privacy	DES AES 3DES SKIPJACK RC4 RC5 RSA El Gamal	Encrypt Decrypt	Primary use of RSA and El Gamal (encryption) is key management
Authenticity & Integrity	DES DSA & SHA RSA & hash El Gamal & hash	Compute and verify MAC (DES) Sign and verify (DSA & SHA, RSA & hash, El Gamal & hash)	
Non-Repudiation	DSS(Digital Signature Standard) RSA Digital Signature ECDSA HECC with DSA	Digital Signature	Minimum Key size 1024 for DSS & RSA Digital Signatures Minimum 160bit key size for ECDSA [56]

Table 6 – Cryptography Services & Protocols

CHAPTER 5 Security Design Engineering

Introduction –

Security Design Engineering is where ambiguities and ideas are translated and transformed into reality, where the what and why of requirements become the who, when, where, and how of the software to be.

From a functional perspective, this transition from desire to actual form is second only to the requirements phase in contributing to the overall quality and success of the eventual software deliverable. From a security perspective, Design Phases is considered by many experts as the single most critical phase of the SDLC. Good decisions made during this phase will not only yield an approach and structure that are more resilient and resistant to attack, but will often also help to prescribe and guide good decisions in later phases such as code and test. Bad decisions made during this phase can lead to design flaws that can never be overcome or resolved by even the most intelligent and disciplined code and test efforts.

Information vulnerabilities cannot be eliminated through the use of any single tool. However, as part of a comprehensive approach to addressing information vulnerabilities, *cryptography* is a powerful tool that can help to assure the confidentiality and integrity of information in transit and in storage and to authenticate the asserted identity of individuals and computer systems.

The Design Phase of the SDLC represents a critical time for identifying and preventing security flaws before they become part of the software. As the connectivity, complexity, and extensibility of software increase, the importance of effectively addressing security concerns as an integral part of the design process will become even more critical. During this phase in the software development effort, architects, designers, and security analysts have an opportunity to ensure that requirements are interpreted appropriately through a security lens and that appropriate security knowledge is leveraged to give the software structure and form in a way that minimizes security risk.

Analogous to the Software Engineering in Design, the design phase of the security engineering helps in taking appropriate design decision for the identified security requirement and threats, specifying what cryptography services & protocol are best to mitigate the identified security threat. In the next section, we will provide a Framework for Security engineering Process Highlighting the Design Phase of the security engineering.

5.1 Security Design Engineering

Analogous to the Design Phase of Software engineering that deals with designing a software structure that realizes the specification , so depending upon the identified security requirements we identify the modules that provides cryptography 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 cryptography services , design structuring & finally design decisions . The figure 7, shows our , overall Security Engineering Process (SEP) highlighting the Design phase of the security engineering.

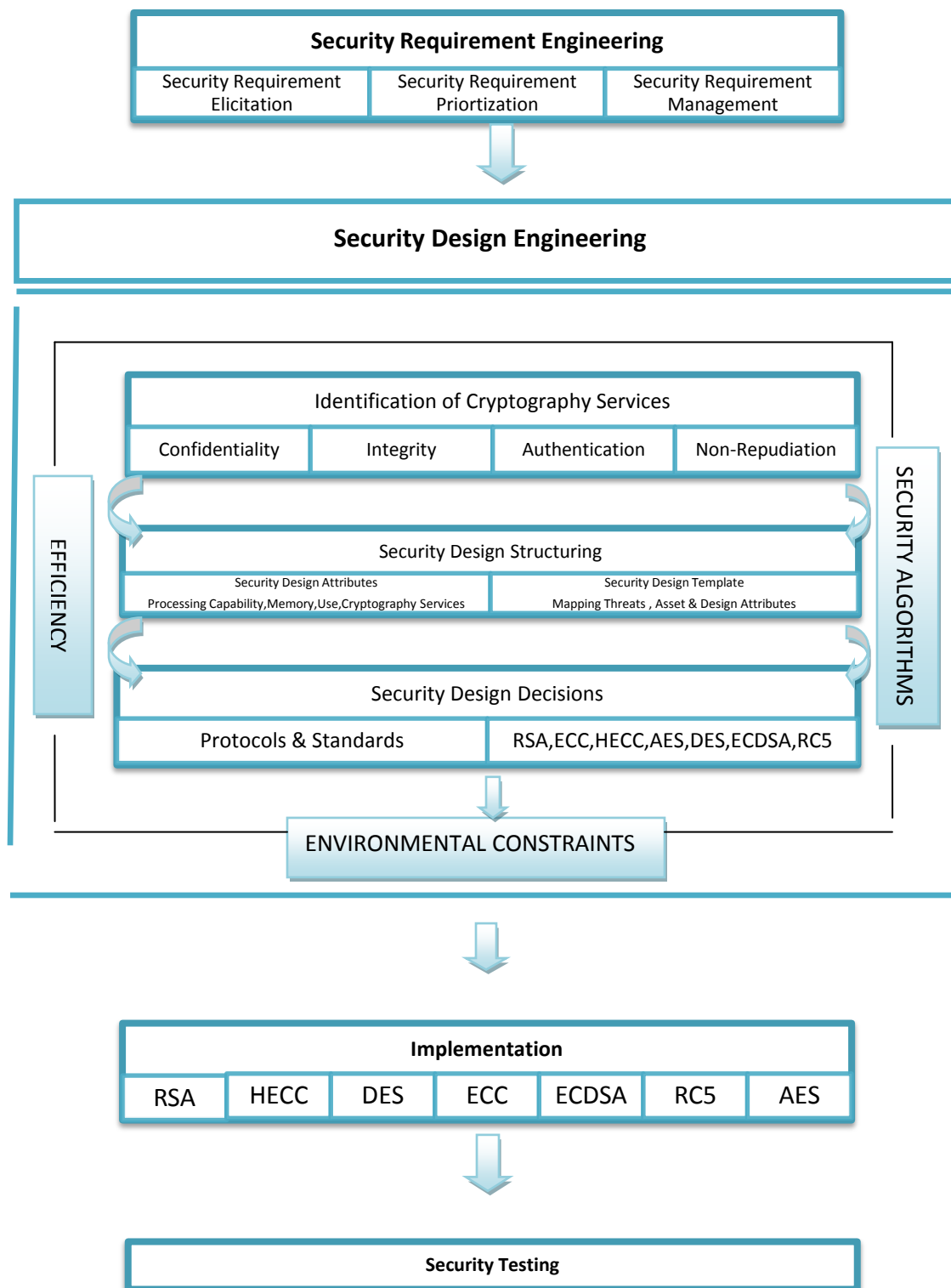


Figure 7 : Framework for Security Engineering Process (SEP) Highlighting the Design Phase

The different activities of the proposed Design Phase in the Security Engineering Process (SEP) are as follows -

1. Identification Of Cryptography Services -

After the security requirements have been identified, we proceed to the design phase of the Security Engineering Process (SEP) i.e. Identification of Cryptography Services. The different types of security requirements proposed by Firesmith [1] are mapped to the different security services provided by cryptography. The various cryptography services, discussed in chapter 4, include Confidentiality services, Integrity services, Authentication services and Non-repudiation services.





Cryptography Services		Security Requirements
Confidentiality		Privacy Requirements
Authentication		Authentication Requirement Authorization Requirement
Non-Repudiation		Identification Requirement Non-Repudiation Requirement
Integrity		Integrity Requirements

Figure 8 : Mapping Cryptography Services to Security Requirements

The identified cryptography services would eventually help in the later stages of the design process , by specifying which cryptography techniques would be usefully in a particular scenario .

For example – As per our finding in table (comparison of different cryptography protocols) in chapter 4 , to satisfy Confidentiality services i.e the privacy requirements , the best available algorithms like DES,AES,3DES could be used . Now , the decision of which cryptography protocol is best suited in the given environment are taken the next

activities depending upon the threat, assets affected , and certain design attributes that will be captured in the next phase i.e design structuring .

So, After the cryptography services have been identified for the particular security requirement, we proceed to the next activity i.e. Security Design Structuring.

2. Security Design Structuring

After the Cryptography services have been identified, we proceed with the following steps -

Step 1 : Identify Security Design Attributes

In this activity , the design attributes are identified that effects the design decisions . These are based on environmental and architectural constraints .The design attributes have been derived upon the different decisive attributes of cryptography protocols like algorithmic complexity ,use scenario , memory constraints & services as explained in chapter 4 .

Following are the design attributes that we will be taking into consideration while making design decisions at the later stage of the Design process.

Attributes
Processor Capabilities
Memory Availability
Use Scenario
Cryptography Service

Table 7 – List of Design Attributes

For Example –

For a particular system, we need take the following things into consideration, as they are deemed important as per our study in the previous chapters.

- What would be the '**Processing Capability**' of the target deployment system. Whether the system would be implemented on a smart card/embedded/Mobile environments or Desktop based systems. This attribute greatly affects our design choices because, only a subset of cryptography algorithms can work efficiently on constrained environments.

- What would be the '**Use Scenario**' of the asset/data (Personal Use / Corporate Use / Critical). The choice of Cryptography Algorithms , differs based upon the asset to be protected . The key-size of the crypto algorithms also forms an important factor in this . For Example –

For protecting Critical Data , RSA with 2048 Bit Key is suggested & for Corporate Level Data , RSA with 1024 bit Key is suggested [34]

- What are the '**Memory**' constraints of the system.

Example –

In a embedded/smart card based environments , the memory constraints are high , therefore careful choices must be made in choosing the algorithm that can work perfectly in such environments .

- What ‘**Cryptography Services**’ would be required. As discussed in activity 1, the Choice of the cryptography algorithm would differ depending upon the service requirement.

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 .

Step 2 : Mapping through Security Design Template

After the security requirement and threats has been identified in the requirement phase & Cryptography Services & 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 for each security requirement .

A Security Design Template contains the values of the design attributes. The values of the design attributes are classified into following categories as shown in the table 8

Design Attributes	Attribute	Rating
	Processing Capabilities	<input type="radio"/> Embedded <input type="radio"/> Desktop
	Memory Availability	<input type="radio"/> Low <input type="radio"/> Medium <input type="radio"/> High
	Use Scenario	<input type="radio"/> Personal <input type="radio"/> Corporate <input type="radio"/> Extremely Valuable Data
	Function	<input type="radio"/> Message Secrecy <input type="radio"/> Message Integrity <input type="radio"/> Authentication /Non repudiation

Table 8 : Values Categories of Design Attributes

For Example – If the security requirements are to be realized on a Smart Cards based environment , that involves access/exchange of Critical data with Confidentiality requirements , then the Design Attributes will have the following values .

Design Attributes	Attribute	Rating
	Processing Capabilities	<input checked="" type="radio"/> Embedded <input type="radio"/> Desktop
	Memory Availability	<input checked="" type="radio"/> Low <input type="radio"/> Medium <input type="radio"/> High
	Use Scenario	<input type="radio"/> Personal <input type="radio"/> Corporate <input checked="" type="radio"/> Extremely Valuable Data
	Cryptography Service	<input checked="" type="radio"/> Message Secrecy <input type="radio"/> Message Integrity <input type="radio"/> Authentication /Non repudiation

Table 9 : Values of Design Attributes

After this , the values of the design attributes are then mapped to the Threat , Assets & Security Requirements , as shown in table 10.

Threat	T.Change_Data	
Asset Affected	Patient Records	
Identified Security Requirement	Privacy Requirement	
Design Attributes	Attribute	Rating
	Processing Capabilities	<input type="radio"/> Embedded <input type="radio"/> Desktop
	Memory Availability	<input type="radio"/> Low <input type="radio"/> Medium <input type="radio"/> High
	Use Scenario	<input type="radio"/> Personal <input type="radio"/> Corporate <input type="radio"/> Extremely Valuable Data
	Function	<input type="radio"/> Message Secrecy <input type="radio"/> Message Integrity <input type="radio"/> Authentication /Non repudiation

Table 10 Security design template for (T.Change_Data , Patient Records , Privacy Requirement)

Step 3 : Security Design Decisions

In this activity , Based upon the extensive Literature Review & Comparison of the efficiency of various security protocols done in chapter 4 , we propose a rule based decision tree approach , to judge the best suitable security protocol depending upon the design attributes supplied in the 'Design Template'.

A Decision tree is constructed based upon the information specified in the design template ,specifying the design path in choosing the optimum security protocol for the system under study. Decision trees are a simple, but powerful form of multiple variable analysis. They provide unique capabilities to supplement, complement, and substitute for recently developed multidimensional forms of reporting and analysis found in the field of business intelligence.

For Example – In the 'Advanced Health Care System ' , one of the identified stakeholder is Specialist doctor which has the facility to access patients records (functional req). The identified threat is T.Change_Data and the asset affected by this threat is patient records.The identified Security requirement in this case is of **Privacy Requirement.**

Now depending upon the information specified in the security design template (For Instance – Target deployment in a *Embedded System with Low Memory constraints*, with desired Message Secrecy / Privacy services for Corporate use).

The identified security requirement i.e. the Privacy requirement can be mitigated by use of cryptography algorithm that enables message secrecy and encryption/decryption of patient records .To find this , we can traverse down the Decisions tree , as shown in figure , that leads us to the best security protocol that will eliminate the T.Change_Data threat on asset Patient records . According to decision tree path shown in figure ,this can be very well done by using

Symmetric algorithms like AES (Advanced Encryption Standard) that enables message secrecy and privacy requirements.

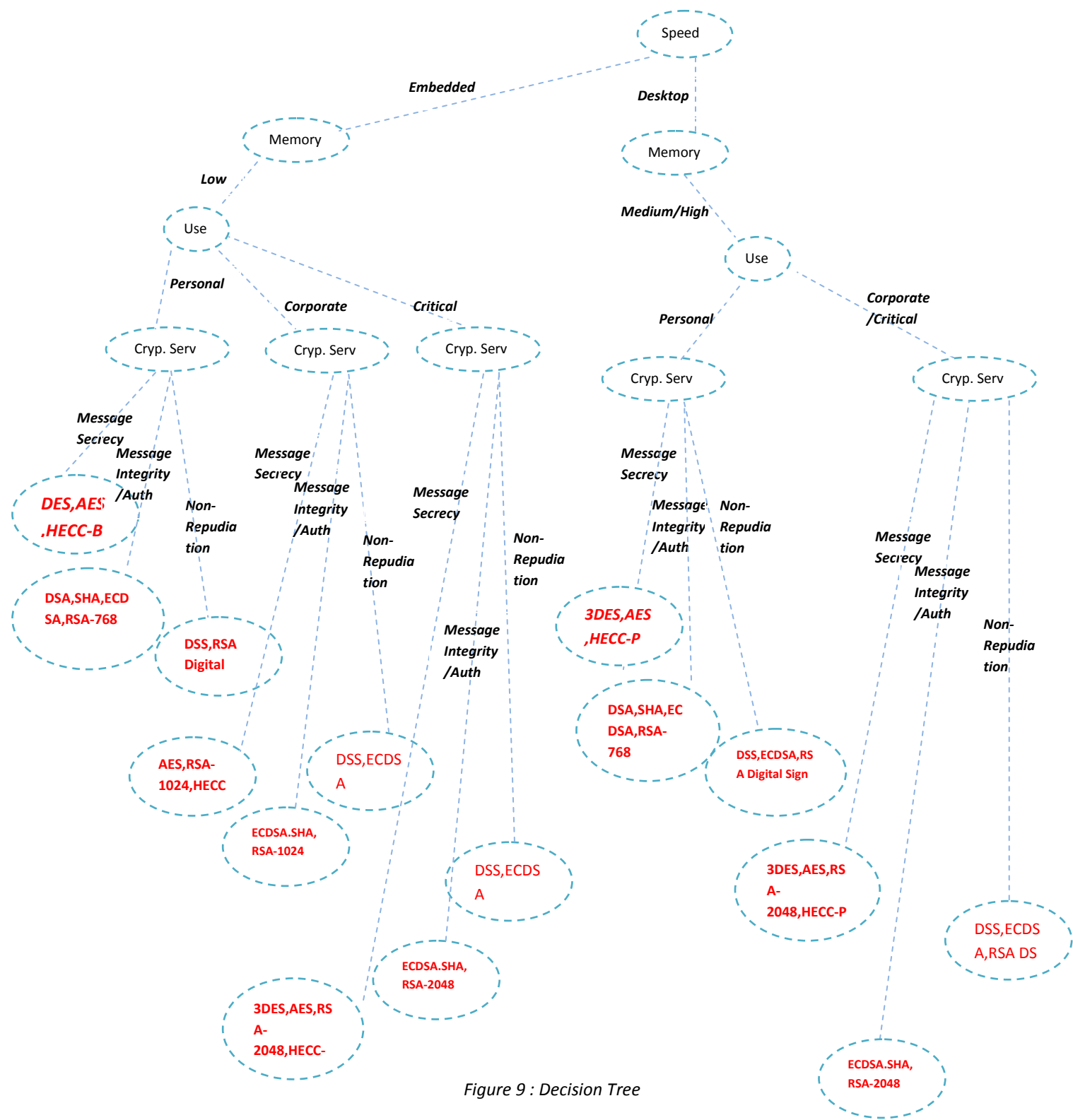


Figure 9 : Decision Tree

CHAPTER 6 Case Studies – Advanced Medical Care System

In this section we go through the development stages of the process defined above using a case study of a CBS system “Advanced Medical Care System” which is partly an online application of normal Hospital Management System to explain our process of security requirement elicitation , prioritization and designing . Advanced Medical Care System is a complete package one needs for a hospital to deal with all the day to day operations taking place. The application can look after Inpatients, OPD patients, records, database treatments, status illness, and billings. It also maintains there in hospital info such as ward ID, Doctor in Charge, and Department administering.

6.1 Requirement Elicitation & Priortization

We starts with the requirements elicitation and analysis for Advanced Medical Care System.

Stage 1 : Identify various stakeholders(actors) – The direct stakeholders of the System would be

6.1.1 The Victim/Patient –

This is the person who is endured in sudden illness or injury.

6.1.2 Paramedic –

He is usually the key person who provides “first aid” in any emergency situation. In any sudden case, this person will interact with the system and try to find out the proper and accurate first aid for any injury or illness.

6.1.3 Hospital Staff (Nurse / Receptionist) –

It will be the hospital staff who is interacting with the system and facilitating the people who want to get “first aid” information, about any sudden case. He or she is also responsible to establish conversation between a specialist doctor and a first aid provider, if he/she failed to get satisfactory material about any special case from the system or he tried the present first aid methods but the victim did not get any pleasing results.

6.1.4 Specialist Doctor –

He is a specialist doctor who is responsible for the medical treatment of any sudden case, or to provide online help to any “first aid” provider.

The indirect stake holders of the system would be System Administrators ,Maintenance Manager & Others . Our interest is only in Direct Actors.

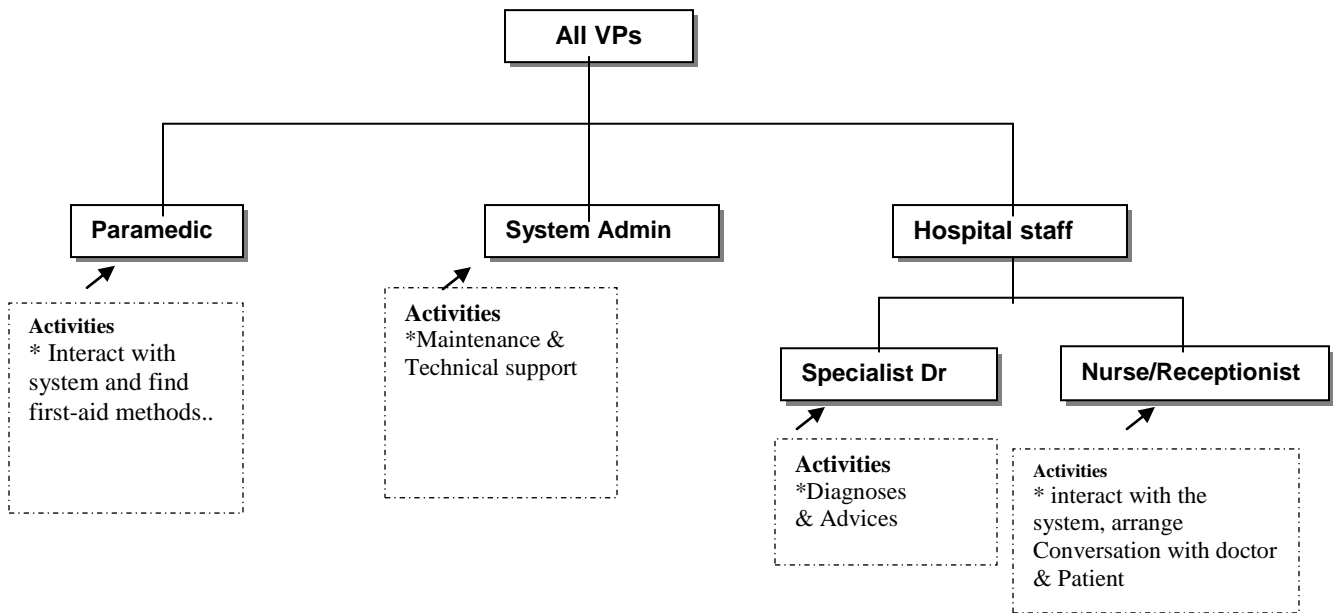


Figure 10 : Viewpoint Hierarchy for Advanced Medical Care System

Stage 2 : Identify Functionality - The functionalities that are required by different stakeholders are as follows.

Functionalities of Paramedic

- **Assign Doctor**
The administrative staff in the ward shall use AMCS to assign a doctor to a given patient. List information about those orders assigned to him/her.
- **Inform Doctors**
The AMCS shall inform doctors of new patients.
- **Initial First Aid**
Responsible for providing First Aid and necessary medical attention initially.

Functionalities of Specialist Dr

- **Emergency Case**
In an emergency case, the administrative staff shall use AMCS to assign an emergency room, doctors and nurses to the patient immediately.
- **Surgery case**
In a surgery case, the administrative staff shall use HPMS to assign a surgery room, surgeon and nurses to the patient.
- **Record procedure**
The whole treatment procedure for the patient shall be recorded by the system.
- **Treat Patient**
- **Recommend Prescription**
- **Access Patients Reports**

Functionalities of Nurse/ Receptionist

- Add patients**

The HPMS shall allow front-desk staff to add new patients to the system.
- Assign Ward**

The consulting nurse shall use AMCS to assign the patient to an appropriate ward.
- Assign ID**

The HPMS shall allow front-desk staff to give each patient a ID and add it to the patient’s record. This ID shall be used by the patient throughout his/her stay in hospital.
- Access Employee Database**

Stage 3 – Identify The Threats associated with each functional requirements or data which is used by this functionality based upon the stakeholders profile. Stakeholders profile has seven fields consisting of -name, functionality, type (as according to view point), Physical location (local or Remote), use case association (read, write, store, update etc.) and whether or not the use case involves exchanging private and secret information. For ex – the Doctor profile is as follows –

Stakeholder	Specialist Doctor
Functionality	Access Patients Reports
Type	Direct
Location	Remote
Private Exchange	True
Secret Exchange	True
Association	Write,Read,Update

Table 11 : Specialist Doctor profile for Access Patient Reports

Now , Threats to the stakeholders is evaluated based upon their profile as follows -

Association = Read	Association = Write	Association = Update
T.Impersonate	T.Change_Data	T.Change_Data
T.Repudiate_Receive	T.Repudiate_Receive	T.Repudiate_Receive
T.Disclose_Data	T.Disclose_Data	T.Disclose_Data
If(Private Exchange = true)	If(Private Exchange = true)	If(Private Exchange = true)
T.Privacy_Violated	T.Privacy_Violated	T.Privacy_Violated
If(Secret Exchange = true)	If(Secret Exchange = true)	If(Secret Exchange = true)
T.Data_Theft	T.Data_Theft	T.Data_Theft
If(Location = remote)	If(Location = local)	If(Location = local)
T.Outsider	T.Insider	T.Insider

Table 12 : Example Showing evaluation of threat based upon Stakeholders profile

Stage 4 : Once the threats has been identified , now we will define security requirements to mitigate these threats . The detailed list of threats and security requirements after performing this step3 and step 4 together is shown below in table.

Viewpoints	Services	Non-functional Requirements	Threats	Security Requirements
Specialist Doctor	1. Treat Patient 2. Surgery Case 3. Emergency Case 4. Access Patients Reports 5. Recommend Prescription 6. Record	1. Reliability with minimum system response time in giving assistance to specialist Doctor 2. Accuracy of information	T.Impersonate T.Data_Theft T.Disclose_Data T.Privacy_Violated T.Change_Data T.Repudiate_Receive	1. Authorization Requirement. 2. Privacy Requirements. 3.. Nonrepudiation Requirements

	Procedure			
Nurse	1. Add Patient 2. Access Employee Database 3. Assign wards 4. Assign Patient Id	1. Correctness of Information. 2. Minimize response time in accessing patients records..	T.Change_Data T.Privacy_Violated T.Social_Engineer T.Outsider	1. Integrity Requirements. 2. Authentication Requirements 3. Identification Requirements

Table 13 – Example of “Advanced Medical Care System” using VOSREP

Stage 5: Analysis & Prioritization of Security Requirements

Once the Security Requirements have been identified , now we will apply the risk analysis process CRAMM [28, 29] for prioritizing security requirements. First of all we will evaluate the risks of the various threats on the assets through CRAMM then based on the measure of risk we will backtrack and prioritize security requirements. The CRAMM process of risk analysis is applied and the output of each phase is shown.

Step 5.1: Identify the Threats & the Assets affected by these threats .

Assets

- Patient Confidential Reports & Information.
 - Employee Information
 - Specialist Doctor & Paramedics Personal Information
 - Authentication System For Employees and Patients Access
- Credit Card Records & E-Commerce Transaction Details
 - Communication Channels with Blood Banks & Other Life Critical Services

Threats

- T.Change_Data
 - T.Repudiate_Receive
 - T.Spoofing
 - T.Flooding
 - T.Disclose_Data
 - T.Privacy_Violated
 - T.Outsider
- T.Integrity
 - T.Data_Theft

Step 5.2 : Identify Potential Asset Impact

THREAT	ASSETS THAT CAN BE AFFECTED
T.Change_Data	Patient Confidential Reports & Information ,Patient Information, Employee Information , E-Commerce Transaction Information
T.Repudiate_Receive	E-Commerce Transaction Information
T.Impersonate	User Login Information , Authentications System for Employees and Patient Access
T.Flooding	Communication Channels with Blood Bank & Other Life Critical Services
T.Disclose_Data	Patient Confidential Reports & Information, employee Information, E-Commerce Transaction Information
T.Privacy_Violated	Patient Confidential Reports & Information, Employee Information
T.Outsider	Credit Card Information
T.Data_Theft	Credit Card & E- Commerce Transaction Information, Patient & Employee Information

Table 14 : Possible Vulnerable Assets

Step 5.3: Value Assets , Threats & Vulnerability

Asset	Value(1 to 10)
Patient Confidential Reports & Information	7
Employee Information	6
Specialist Doctor & Paramedics Personal Information	6
Authentication System For Employees and Patient Access	8
Credit Card Records & E-Commerce Transaction Details	9
Communication Channels with Blood Bank & other Life Critical Services	10

Table 15 : Measure of Asset

Threat	Level Of Threat	Value (.1,.34,1,3.33,10)
T.Change_Data	High	3.33
T.Repudiate_Receive	High	3.33
T.Impersonate	Medium	1
T.Flooding	Very Low	.1
T.Disclose_Data	Medium	1
T.Privacy_Violated	Low	.34
T.Outsider	High	3.33

T.Integrity	Very High	10
T.Data_Theft	Medium	1

Table 16 : Measure of Threat

Threat	Level Of Vulnerability	Value (.1,.5,1)
T.Change_Data	High	1
T.Repudiate_Receive	High	1
T.Impersonate	Medium	.5
T.Flooding	Low	.1
T.Disclose_Data	Medium	.5
T.Privacy_Violated	Low	.1
T.Outsider	High	1
T.Integrity	High	1
T.Data_Theft	Medium	.5

Table 17 : Level of Vulnerability

Step 5.4: Estimate Risk

Once we have calculated the value of threats and assets we will use the Three – Dimension lookup table to measure the level of risk given by CRAMM. where the strength of the threat, the level of the vulnerability and the value of the asset are input parameters, gives the final security requirement (= risk) in the range 1 through 7.

Security Requirement	Threat	Threat Rating	Vulnerability	Asset Affected (Threat)	Asset Value	Risk
Authorization Requirement	1)T.Impersonate	1	.5	Patient Confidential Reports & Information (4,3)	7	6,5
	2)T.Data_Theft	1	.5			
	3)T.Disclose_Data	1	.5	Employee Information (4,2,3)	6	5,4,4
	4)T.Change_Data	3.33	1	E-Commerce Transaction Information(4)	9	7
				User Login Information (1)	9	
				Authentication System (1)	8	6
				Credit Card Trx Info (2,3)	9	5
Privacy Requirement	T.Change_Data	1	.5	Patient Records(4)	7	6

Table 18 : Measurement of Risk Levels

6.2 Security Design Phase

After the Security requirements have been prioritized , we proceed to the design phase where design decisions are identified for each security requirement , in the order of their priority levels .

The different activities performed during the Design Phase in the Security Engineering Process (SEP) are as follows –

Step 1 : Identification Of Cryptography Services -

At this step , the identified security requirement is mapped to the appropriate cryptography services as discussed in 5.1 .

From the Table 18 , we take up the privacy security requirement first , as it has the highest priority .

Security Requirement	Threat	Threat Rating	Vulnerability	Asset Affected (Threat)	Asset Value	Risk
Privacy Requirement	T.Change_Data	1	.5	Patient Records(4)	7	6

Table 19 : Security Requirement , Threat , Asset , Risk Table

Here the identified security requirement is of Privacy requirement , which maps to the Confidentiality services of the cryptography .

Security Requirements	Cryptography Services
Privacy Requirements	Confidentiality Service

Figure 11 : Mapping Security Requirements to Cryptography Services

So , we prepare a table showing the Actor , Asset , threat ,functional requirement & security requirements mapping to the desired cryptography services.

Viewpoints	Functional Requirement	Asset	Threats	Security Requirements	Cryptography Services
Specialist Doctor	Access Patients Reports	Patient Records	T.Change_Data	Privacy Requirements.	Confidentiality Services

Table 20 : Mapping Cryptography Services to Viewpoint , Asset , Threat & Security Requirement

So, After the cryptography services have been identified for the particular security requirement, we proceed to the next activity i.e. Security Design Structuring.

Step 2 : Security Design Structuring –

I. Identify Security Design Attributes

In this activity , the design attributes are identified that effects the design decisions as described in chapter 5 .

Attribute
Processor Capabilities
Memory Availability
Use Scenario
Function

Table 21 - Design Attributes

For the Functional Requirement ‘Access patient records ‘ , the viewpoint (i.e. the specialist doctor) will be using a target mobile platform for accessing patient records .The values for all these attributes can be derived from already captured non functional requirement & environmental constraints that are captured during the requirement phase .

II. Mapping through Security Design Template

A Security Design Template contains the values of the design attributes providing a mapping between the assets, threats & security requirements.

In our system , the Module ‘Access patient records ‘ will be realized on a Smart Cards based environment , that involves access/exchange of Critical data with Confidentiality requirements , then the Design Attributes will have the following values .

Design Attributes	Attribute	Rating
	Processing Capabilities	<input checked="" type="radio"/> Embedded <input type="radio"/> Desktop
	Memory Availability	<input checked="" type="radio"/> Low <input type="radio"/> Medium <input type="radio"/> High
	Use Scenario	<input checked="" type="radio"/> Personal <input type="radio"/> Corporate <input type="radio"/> Extremely Valuable Data
	Cryptography Service	<input checked="" type="radio"/> Message Secrecy <input type="radio"/> Message Integrity <input type="radio"/> Authentication /Non repudiation

Table 22 : Values of Design Attributes for Threat T.Change_Data on Asset Patient Records with Privacy Requirements

After this , the values of the design attributes are then mapped to the Threat , Assets & Security Requirements , as shown in table 23.

Threat	T.Change_Data	
Asset Affected	Patient Records	
Identified Security Requirement	Privacy Requirement	
Design Attributes	Attribute	Rating
	Processing Capabilities	<input checked="" type="radio"/> Embedded <input type="radio"/> Desktop
	Memory Availability	<input checked="" type="radio"/> Low <input type="radio"/> Medium <input type="radio"/> High
	Use Scenario	<input checked="" type="radio"/> Personal <input type="radio"/> Corporate <input type="radio"/> Extremely Valuable Data
	Function	<input checked="" type="radio"/> Message Secrecy <input type="radio"/> Message Integrity <input type="radio"/> Authentication /Non repudiation

Table 23 Security design template for (T.Change_Data , Patient Records , Privacy Requirement)

III. In this activity , we will identify the set of security protocol that mitigates the identified security threat using the decision tree based approach. A Decision tree is constructed based upon the input design template that specifies the design path in choosing the optimum security protocol for the system under study.

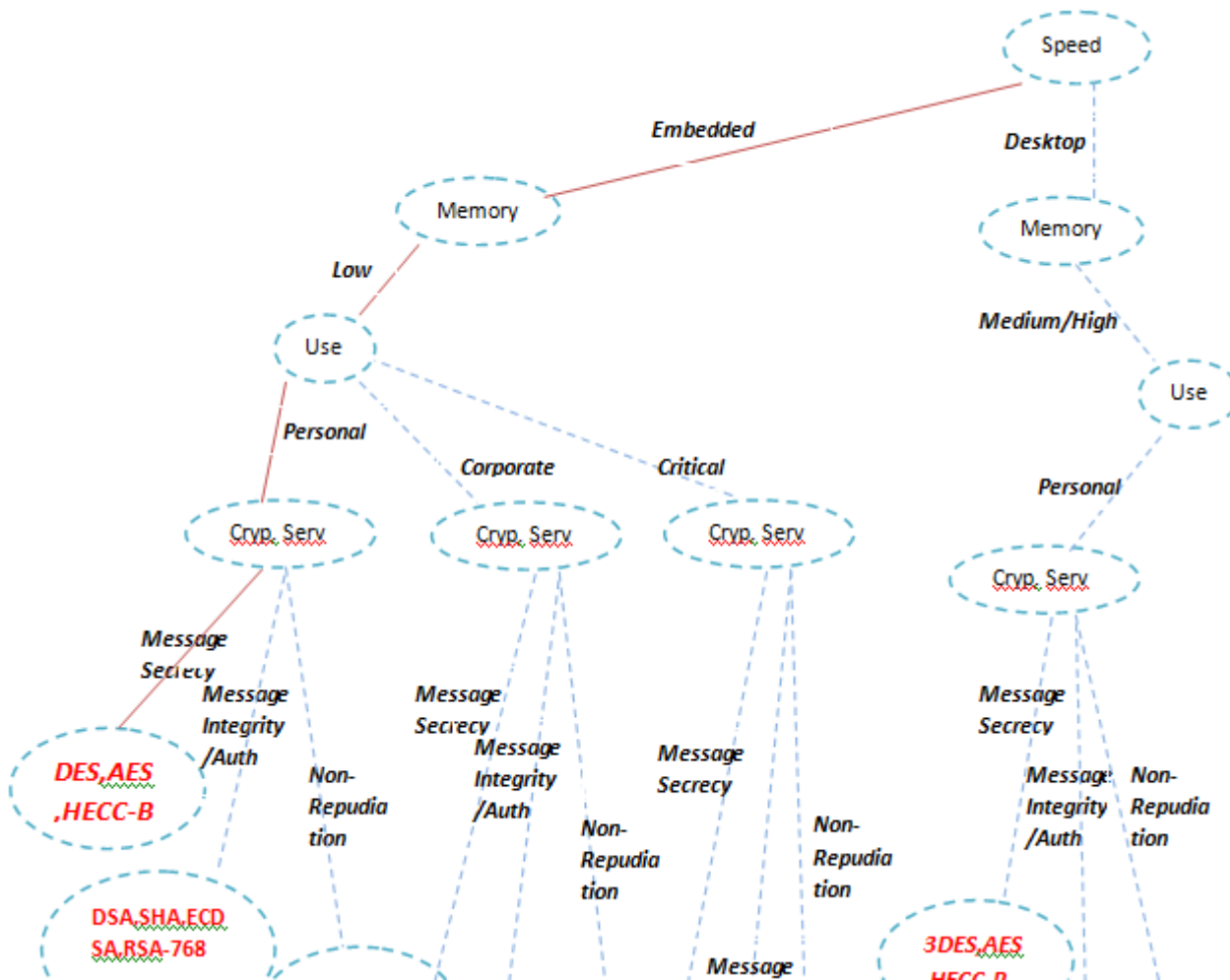


Figure 12 : Decision Tree Path for Privacy Requirement Leading us to the best cryptography protocol.

In the ‘Advanced Health Care System ‘, one of the identified stakeholder is Specialist doctor which has the facility to access patients records (functional req). The identified threat is T.Change_Data and the asset affected by this threat is patient records.The identified Security requirement in this case is of **Privacy Requirement**.

Now depending upon the information specified in the security design template of table 20 that specifies that the Target deployment is a *Embedded System(Mobile)* with *Low Memory constraints*, with desired Message Secrecy / Privacy services for personal data and the identified identified security requirement i.e. the Privacy requirement can be mitigated by use of cryptography algorithm that enables message secrecy and encryption/decryption of patient records .To find this , we can traverse down the Decisions tree , as shown in figure 12 , that leads us to the best security protocol that will eliminate the T.Change_Data threat on asset Patient records . According to decision tree path shown in figure 12 ,this can be very well done by using Symmetric algorithms like AES (Advanced Encryption Standard) ,DES (Data Encryption Standard) , HECC-B (Hyperelliptic Curve over binary fields) that enables message secrecy and privacy requirements.

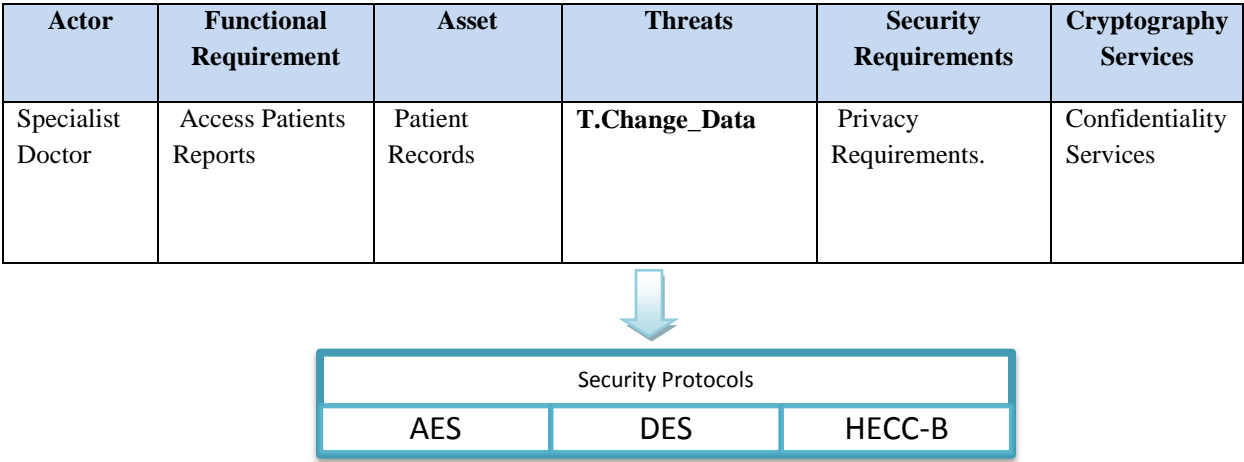


Figure 13 : Security design Decisions

6.3 ATM System Example -

Now , we will take an another Example of an ‘Automated Teller Machine (ATM)’ focusing on the design related activities , to make the process more clear . In this example , we will restrict our study to only one identified security requirement .

Step 1 : Identify Security Requirements & Prioritize –

An ATM Machine keeps record of all the user transactions and their ATM/credit card numbers. Now , this information must be protected from malice actors or entities .The malice actor could be a User or a Machine maintenance staff who has knowledge and access to internal subsystem of the ATM machine.

So in this case , the identified viewpoint is Customer that issues a transaction on ATM Machine , the functional requirement would be a withdrawal transaction , the asset affected would be ATM& Credit card details which are stored as transaction logs within the ATM system memory. The identified threat would be T.Data_Theft , So our security requirement associated with this asset and threat would be Privacy Requirement .

Viewpoints	Functional Requirement	Asset	Threats	Security Requirements
Customer	Withdraw Transaction	ATM/Debit/ Credit Card Information	T.Data_Theft	Privacy Requirements.

Table 26 : One of the Identified Security requirement for an ATM System

Since , we are considering only one security requirement in this example , we have chosen the security requirement which has the higher risk priory .

Step 2: Design Phase

After the security requirement have been identified , we proceed to the design phase

Step 2.1 – Identify Cryptography Services –

The user transaction details and card information must be kept in a protected form outside the reach of hackers or harmful actors like a machine maintenance staff who has the knowledge of the various H/w of the ATM Machine & could easily access the data from the storage devices if the confidentiality of the transaction data is not maintained.

So, here the identified security requirement is of Privacy requirement, which maps to the Confidentiality services of the cryptography.

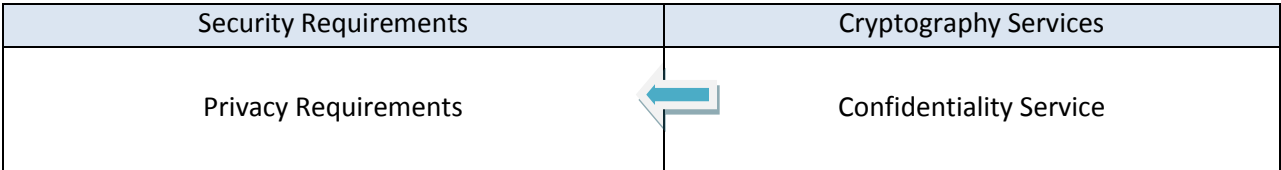


Figure 19 : Mapping Security Requirements to Cryptography Services for ATM System

So, we prepare a table showing the Actor, Asset , threat ,functional requirement & security requirements mapping to the desired cryptography services.

Viewpoints	Functional Requirement	Asset	Threats	Security Requirements	Cryptography Services
Customer	Withdraw Transaction	ATM/Debit/ Credit Card Information	T.Data_Theft	Privacy Requirements.	Confidentiality Services

Table 27 : Mapping Cryptography Services to Viewpoint , Asset , Threat & Security Requirement for ATM System

So, After the cryptography services have been identified for the particular security requirement, we proceed to the next activity i.e. Security Design Structuring.

Step 2.2 : Design Structuring

Now to mitigate the identified threat associated with the asset (credit/atm/debit card information) for privacy security requirement, we will proceed with the following steps

Step 2.2.1 – Identify Design Attributes –

We will take up the standard design attributes described in chapter 5 for this analysis

Step 2.2.2 – Security Design Template –

In this activity the values of the design attributes are taken from a database where environmental and non functional requirements are gathered .Taking an example of Model 8100 automated teller machine , the design attributes would have the following values . A general ATM Machine would consist of Intel Celeron (P4) 2000 MHz processor with 512 MB or higher operating memory and 40 Gb hard disk and RSA 232 Port for external communication .

Design Attributes	Attribute	Rating		
	Processing Capabilities	<input type="radio"/> Embedded	<input checked="" type="radio"/> Desktop	
	Memory Availability	<input type="radio"/> Low	<input checked="" type="radio"/> Medium	<input type="radio"/> High
	Use Scenario	<input type="radio"/> Personal	<input type="radio"/> Corporate	<input checked="" type="radio"/> Extremely Valuable Data
	Cryptography Service	<input checked="" type="radio"/> Message Secrecy	<input type="radio"/> Message Integrity	<input type="radio"/> Authentication /Non repudiation

Table 28 : Values of Design Attributes for Threat T.Data_Theft on Credit Card / ATM Information with Privacy Requirements

After this , the values of the design attributes are then mapped to the Threat , Assets & Security Requirements , as shown in table 29.

Threat	T.Data_ Theft		
Asset Affected	ATM/Debit/Credit Card Details		
Identified Security Requirement	Privacy Requirement		
Design Attributes	Attribute	Rating	
	Processing Capabilities	<input type="radio"/> Embedded	<input checked="" type="radio"/> Desktop
	Memory Availability	<input type="radio"/> Low	<input checked="" type="radio"/> Medium <input type="radio"/> High
	Use Scenario	<input type="radio"/> Personal	<input type="radio"/> Corporate <input checked="" type="radio"/> Extremely Valuable Data
	Function	<input checked="" type="radio"/> Message Secrecy	<input type="radio"/> Message Integrity <input type="radio"/> Authentication /Non repudiation

Table 29 Security design template for (T.Data_ Theft , ATM/Debit/Credit card , Privacy Requirement)

Step 2.2.3 : Design Decision

At the end, based upon the Security Design template, we will traverse the decision tree given in chapter 5 , to identify the set of best protocols that mitigates the identified security requirement .

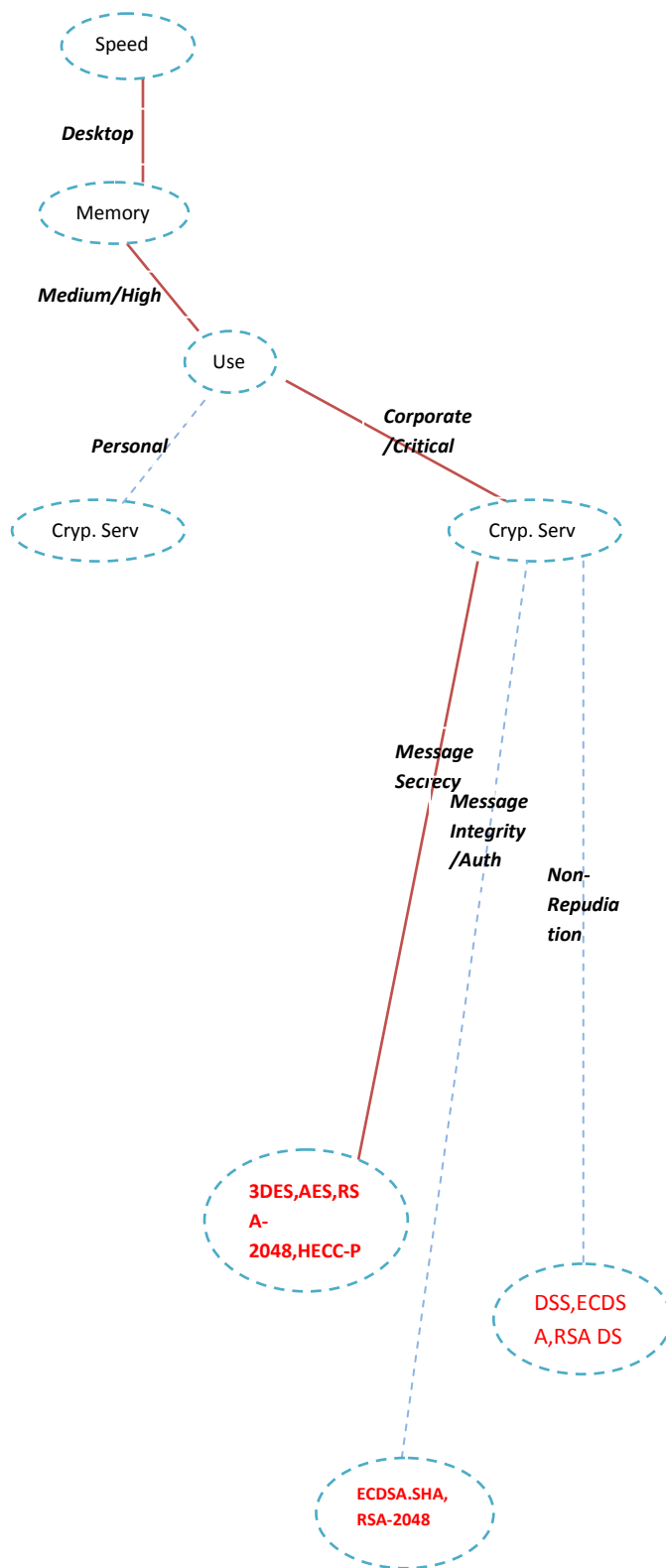


Figure 21 : Decision Tree for Design Decisions for ATM System

To find this , we can traverse down the Decisions tree , as shown in figure , that leads us to the best security protocol that will eliminate the T.Data_Theft threat on asset ATM/Debit/Credit card information . According to decision tree path shown in figure ,this can be very well done by using Symmetric algorithms like AES (Advanced Encryption Standard) ,3DES (triple Data Encryption Standard) , HECC-P (Hyperelliptic Curve over prime fields), RSA 2048 Bit key that enables message secrecy and privacy requirements.

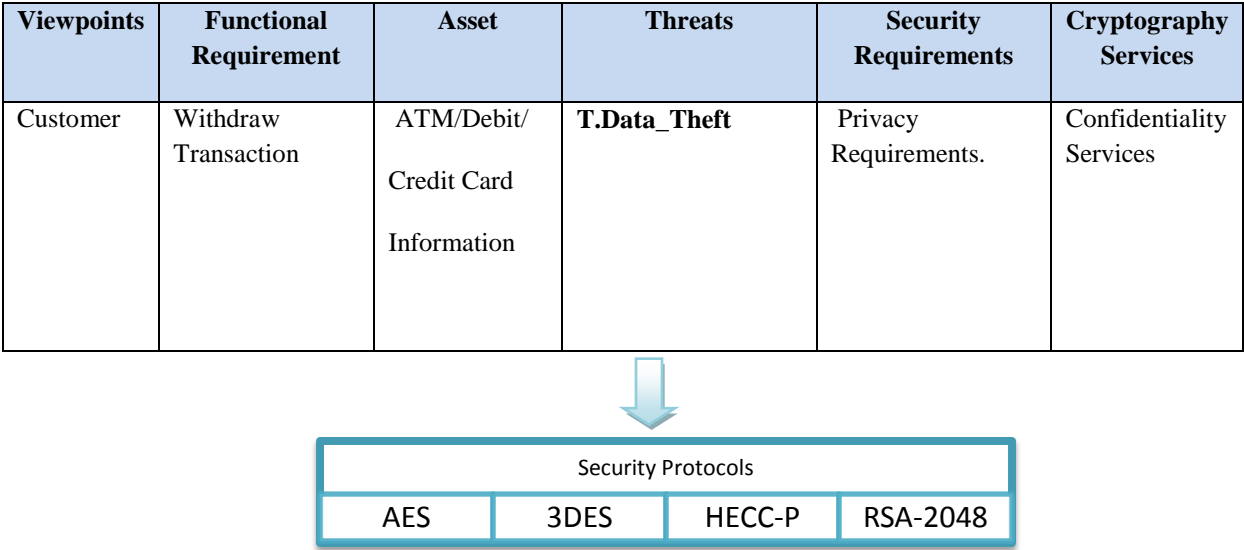


Figure 20 : Security design Decisions for Privacy requirement of ATM System

CHAPTER 7 Conclusion

The work carried out in this report provides a strong base in Security Design Engineering taking into the perspectives of cryptography. From the very initial stage where security requirements are elicited and prioritized, we have proposed activities that helps in taking the design decision, thereby mitigating the identified security threat.

A comprehensive study of the various parameters of security protocols allows us to make accurate design decisions thereby increasing the efficiency of our design process. Our main emphasis is on presenting the techniques for identification of cryptography services that are mapped to the security requirements , then security design structuring where security design attributes are identified and mapped to threats, assets & security requirements .Lastly , converting these into design decisions.

For the future work , we have proposed a framework that seamlessly integrates all the cryptography algorithms & in real time , calculates it suitability and performance based upon the design attributes & environmental constraints . This system would take into account more rigorous security design attributes from the cryptography perspective and would work on the lines of a DSS (Decision support system) that would help the designers in making appropriate design choices , thereby increasing the efficiency of the system being developed.

References

- [1] Donald G. Firesmith, "Engineering Security Requirements", Journal of object technology, 2003, vol 2, no.1, pages 53-68.
- [2] CERT (www.cert.org) , Carnegie Mellon University.
- [3] Patterns & Practices Security Engineering Index, J.D. Meier, Alex Mackman, Blaine Wastell, Prashant Bansode, Kishore Gopalan [2005] , Microsoft Corporation
<http://msdn.microsoft.com/en-us/library/ff648032.aspx>
- [4] Kotonya G., Sommerville I., "Requirement Engineering with view points", 1995.
- [5] Sommerville, I., "Software Engineering". Seventh edition 2003. ISBN - 8129708671. Pearson Education.
- [6] Gupta D., Agarwal A., "Security Requirement Elicitation using view points for online system", International Conference on Emerging Trends in Engineering and Technology, Nagpur, July 2008.
- [7] Caralli, Richard," Sustaining Operational Resiliency: A Process Improvement Approach to Security Management." Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.
<http://www.sei.cmu.edu/publications/documents/06.reports/06tn009.html>
- [8] CWE 2007 , MITRE Corporation " Common Weakness Enumeration " , <http://cwe.mitre.org> (2007).
- [9] Jones, Capers " Programming Productivity " New York: McGraw-Hill, 1986.
- [10] Boehm, Barry W., & Papaccio, Philip N. "Understanding and Controlling Software Costs. IEEE Transactions on Software Engineering 14, 10 (October 1988)

- [11] Donald G. Firesmith, "Security Use cases", Journal of object technology, 2003, vol 2, no.3, pages 53-64.
- [12] John Mc Dermott, Chris Fox, "Using abuse case models for security requirements analysis." Department of Computer Science, James Madison University, 1999.
- [13] Alexander IF, "Misuse cases, use cases with hostile intent". IEEE Software, 2003, pages 58–66
- [14] Common criteria for information technology security evaluation. Technical report CCIMB 99–031, Common Criteria Implementation Board, 1999.
- [15] M. Ware, J. Bowles, C. Eastman, "Using the common criteria to Elicit security Requirements with use cases", 2006 IEEE Computer Society.
- [16] Robert J. Ellison, "Attack Trees" Software Engineering Institute, Carnegie Mellon University, 2005.
- [17] EBIOS- Expression of need and identification of security objectives , DCSSI, France, February,2004.
- [18] Alberts, Christopher and Dorofee, Audrey. OCTAVE Method Implementation Guide v2.0. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2001.
<http://www.cert.org/octave>
- [19] CORAS - <http://www2.nr.no/coras>.
- [20] The Logic behind CRAMM's Assessment of Measures of Risk and Determination of Appropriate Countermeasures , www.cramm.com
- [21] USE OF THE ZACHMAN ARCHITECTURE FOR SECURITY ENGINEERING ,Ronda R. Henning ,Harris Corporation , Information Systems Division
- [22] Kotonya G., Sommerville I., "Requirement Engineering with view points", 1995.

- [23] Donald G. Firesmith, "Engineering Security Requirements", Journal of object technology, 2003, Vol2, no.1, pages 53-68.
- [24] Sindre, Guttorm, & Opdahl, Andreas L. "Eliciting Security Requirements by Misuse Cases," 120–131. Proceedings of the 37th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS-37 '00). New York: IEEE Press, 2000.
- [25] Checkland, Peter. Soft Systems Methodology in Action. Toronto, Ontario, Canada: John Wiley & Sons, 1990.
- [26] QFD Institute. Frequently Asked Questions About QFD.
http://www.qfdi.org/what_is_qfd/faqs_about_qfd.htm (2005).
- [27] Christel, M., & Kang, K. Issues in Requirements Elicitation (CMU/SEI-92-TR-012, ADA258932). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1992.
<http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.012.html>
- [28] Kunz, Werner, & Rittel, Horst. "Issues as Elements of Information Systems." <http://www-iurd.ced.berkeley.edu/pub/WP-131.pdf> (1970)
- [29] Schiffrin, D. Approaches to Discourse. Oxford, UK: Blackwell, 1994.
- [30] John Mc Dermott, Chris Fox, "Using abuse case models for security requirements analysis." Department of Computer Science, James Madison University, 1999.
- [35] Ian Hawkins, "Risk Analysis Techniques", 1998
<http://www.euclidresearch.com/current.html>
- [36] W.G. Bornman, L. Labuschagne, "A Comparative Framework for Evaluating Information Security Risk Management Methods", (icsa.cs.up.ac.za/issa/2004/Proceedings/Full/015.pdf)
- [33] O. Kömmerling and M. G. KuhnDesign, "Principles for tamperresistant smartcard processors," presented at the USENIX Workshop Smartcard Technology (Smartcard '99), Chicago, IL.

- [34] P. C. Kocher, "Timing attacks on implementations of Diffie–Hellman, RSA, DSS, and other systems," in *Advances in Cryptology—CRYPTO'96*, N. Koblitz, Ed.. Heidelberg, Germany: Springer-Verlag, 1996, vol. 1109, *Lecture Notes in Computer Science*, pp. 104–113.
- [37] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology—CRYPTO'99*, M. Wiener, Ed. Heidelberg, Germany: Springer-Verlag, 1999, vol. 1666, *Lecture Notes in Computer Science*, pp. 388–397
- [38] I. Biehl, B. Meyer, and V. Muller, "Differential fault attacks on elliptic curve cryptosystems," in *Advances in Cryptology—CRYPTO 2000*, M. Bellare, Ed. Heidelberg, Germany: Springer-Verlag, 2000, vol. 1880, *Lecture Notes in Computer Science*, pp. 131–146
- [39] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic analysis: Concrete results," in *Cryptographic Hardware and Embedded Systems—CHES 2001*, Ç. K. Koç, D. Naccache, and C. Paar, Eds. Heidelberg, Germany: Springer-Verlag, 2001, vol. 2162, *Lecture Notes in Computer Science*, pp. 251–261
- [40] NIST FIPS PUB 46-3 . Data Encryption Standard . Federal Information Processing Standards , National Bureau of Standards , US Department of Commerce, 1977
- [41] K. W. Campbell and M.J. Wiener, "Proof that DES is Not a Group," *Advances in Cryptology - CRYPTO '92*
- [42] *Types of Cryptographic Attacks* , Eric Conrad , 2007
- [43] *Computational Complexity of Cryptanalysis Problems* , National seminar on e-security education through e-learning,e-learn '2007 , 14 December 2007 , C-DAC, Noida C.E. Veni Madhavan <http://homes.esat.kuleuven.be/~fvercaut/talks/ECDL.pdf>
- [44] *Elliptic curve discrete logarithm problem* , Dr. F. Vercvaeteren <http://homes.esat.kuleuven.be/~fvercaut/talks/ECDL.pdf>

- [45] RSA Problem , Ronald L. Rivest, MIT Laboratory for Computer Science - <http://people.csail.mit.edu/rivest/RivestKaliski-RSAProblem.pdf>
- [46] Niels Ferguson , Bruce Schneier ,” Cryptography Engineering: Design Principles and Practical Applications “
- [47] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. Handbook of Applied Cryptography. CRC Press, Boca Raton, Florida, USA, 1997. Chapter 14
- [48] B. Schneier “ Applied Cryptography “ John Wiley & Sons Inc., New York, New York, USA, 2nd edition, 1996.
- [49] R. L. Rivest, A. Shamir, and L. Adleman “ A Method for Obtaining Digital Signatures and Public-Key Cryptosystems “ Communications of the ACM, 21(2):120{126, February 1978.
- [50] P. Barrett ,” Implementing the Rivest, Shamir and Adleman public-key encryption algorithm on a standard digital signal processor” , Advances in Cryptology: CRYPTO’86 (A. M. Odlyzko ed.), LCNS 263, Springer-Verlag, pp. 311-323, 1987.
- [51] P.H. Hartel, P. Paradinas, and J.-J. Quisquater, “ Arithmetic Co-processors for Public-key Cryptography: The State of the Art “ 1996.
- [52] Sung-ho Jee and Paul Montague, “ A Secure DES Implementation for Real-time Embedded Applications”, Motorola Australia Software Centre, Adelaide, SA 5095, Australia
- [53] Norman D. Jorstad , “CRYPTOGRAPHIC ALGORITHM METRICS “ 2007
- [54] Thomas Wollinger, Jorge Guajardo, and Christof Paar , “Cryptography in Embedded Systems: An Overview “ , Proceedings of the Embedded World 2003 Exhibition and Conference, pp. 735-744, Design & Elektronik, Nuernberg, Germany, February 18-20, 2003.
- [55] Eran Rippel , “Security Challenges in Embedded Designs “ , Discretix Technologies, Ltd. <http://www.design-reuse.com/articles/20671/security-embedded-design.html>

- [56] "Authentication/Integrity Algorithm Issues Survey" , CCSDS 350.3-G-1, march 2008
<http://public.ccsds.org/publications/archive/350x3g1.pdf>
- [57] N. Mayer, P. Heymans, R. Matulevičius "Design of a Modelling Language for Information System Security Risk Management", In Proceedings of the First International Conference RCIS – 2007.
- [58] Fernandes, A."Elliptic Curve Cryptography." *Dr. Dobb's Journal*, December 1999.
- [59] Biehl et. al. ,"[Differential Fault Attacks on Elliptic Curve Cryptosystems](http://www.iacr.org/archive/crypto2000/18800131/18800131.pdf)".
<http://www.iacr.org/archive/crypto2000/18800131/18800131.pdf>.
- [60] Koblitz , ' Hyperelliptic cryptography ' , journal of Cryptography , 1989
- [61] Thomas Wollinger , "Elliptic & Hyperelliptic Curves on Embedded Micro Processor", ACM,March 2003
- [62]Jan Pelzl , " Hyperelliptic Curve cryptosystem : Closing the performance gap to elliptic curves " , springer 2003