Secure Software Framework for Process Improvement

Dr.R.Surendiran
Assistant Professor
Mass College of Arts & Science, Kumbakonam

Abstract:
Nowadays, Web applications are developing quickly.Software security has become an excessive challenge as the ratio of breaches is growing. The highest purpose behind the security breaches of software systems is the lack of security concern throughout the quick development stages. There are two views in software development, Product view and process view. Previous is concerned is what is to be developed and the latter is concerned with how it is to be established. In this paper we explain how the lack of concentration on security in software process improvement in to security vulnerabilities, and we propose an agile method for safe software design that needs team members to have received suitable security education and training.

Keywords: *Security, Secure software Life cycle, Software systems, architectural, development.*

I INTRODUCTION

Developing secure software is not an benefit but has developed an essential for the software organization. In now networked environment, the software is becoming extra vulnerable to both the thoughtful and accidental malicious intentions. The central purpose behind the security holes in the software is due to the abandon of addressing security problems in the software development method. Security is continuously preserved as an addition in the software development process, and dispersed with next the system development phases by providing the compulsory defensive measures. Security weaknesses in software are classically produced by programmers or groups with insufficient expertise in secure software development. Inappropriately, thousands of IT originators and specialists about the domain are missing security skills exactly because cyber security was not extent of the education program they tracked at the university.

The increasing ICT infrastructure worldwide is existence built by IT specialists with IT degrees from universities and colleges, but regrettably many IT specialists still have inadequate security thoughtful and proficiency. This is an unacceptable situation. Software security is near engineer software in such a way that thedesired request purpose uninterrupted and is capable to correctly handle the security extortions over malicious attacks.

Security approves that application all in a selected manner and to offer defense against safety threats. In mutual practice, security is ignored in initial stages of software development life cycle (SDLC).

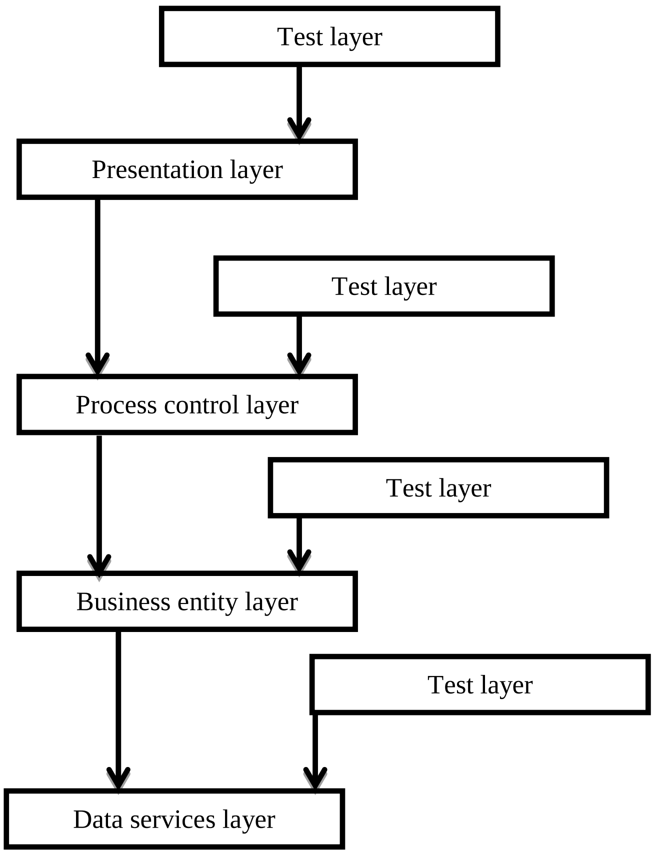


Fig:1 Agile Software Architecture

1. Agile development in security

Agile development Security analysis of Web-based Systems via hypertext transfer Protocol Part is whole by hiring a extremely testable manner and consuming an automatic analysis outline. The framework can avoid the presentation layers and interconnect straight with the essential network application server via hypertext transfer protocol Part. This offers the group of capability to achieve security testing for dangerous susceptibilities that are finest lessened by secure programming achieves on the network application server Agile Security Testing

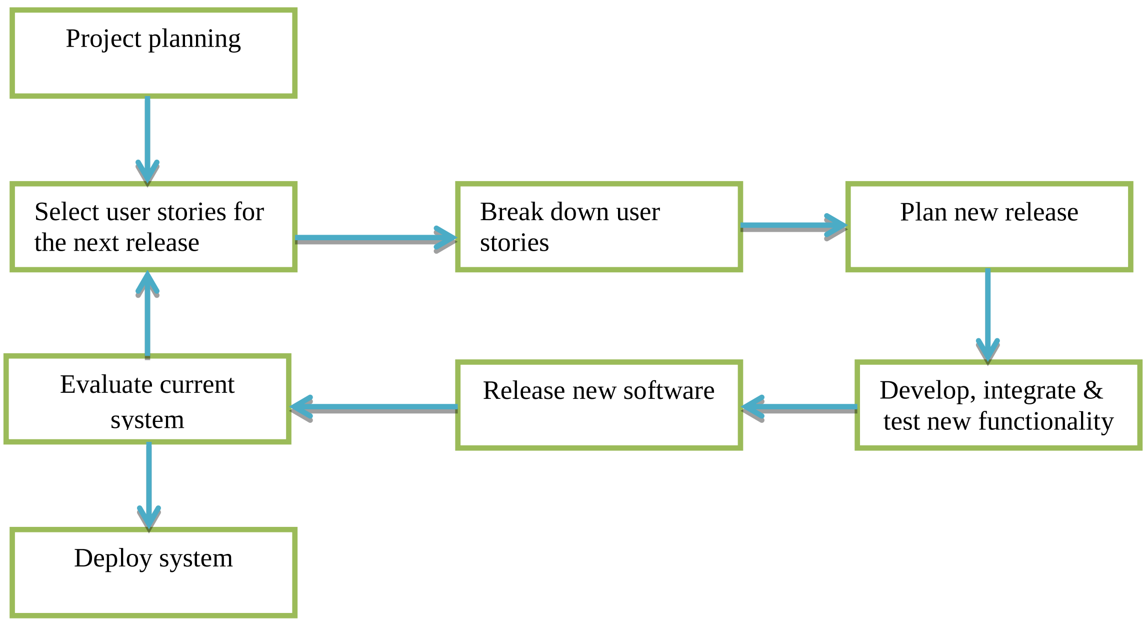
2. Diffusion testing and mitigation false positive:
After each execution of penetration testing, reviews the result and detect false positive. This needs tools to mark false positive and preventing them in the next execution.

3. Investigation evaluation:
It shows the reason that why some bugs are not discovered in development phase and patching the test tools for cover them.

method is protracted, and three additional phases are augmented to it as,

4. Information repository: It is for redeemable some respected information. The novel technique has called Extended Agile Security Testing it is described as few steps

1. Design misuse case.
2. Use testable layer architecture.
3. Automatic code review.
4. Fill knowledge repository.
5. Penetration testing and mitigation false positive.
6. Postmortem evaluation.



.

Fig: 2 the agile model for software development

IIPROPOSED METHOD:

Our goal with the software security framework is to capture a complete high-level understanding that comprises all of the important software security initiatives. Note that separately these initiatives follow dissimilar procedures

1 Governance
2 Intelligence
3 SDL Touch points
4 Deployment

1. Governance
Those performs that support establish, manage, and measure a software security initiative. Staff improvement is also a significant governance practice.

❖ ***Approach and metrics:***
Practice encompasses planning, assigning parts and responsibilities, detecting software security aims, determining finances, and identifying metrics and entries.

❖ ***Training:***
Has continuously played a grave role in software security since software developers and designers frequently start with very slight security knowledge.

2. Intelligence
Performs that consequence in groups of business knowledge used in carrying out software security actions throughout the group. Collections comprise together proactive safety guidance and organizational threat modeling. The cleverness domain is to make organization-wide possessions. Those possessions are divided into three practices.

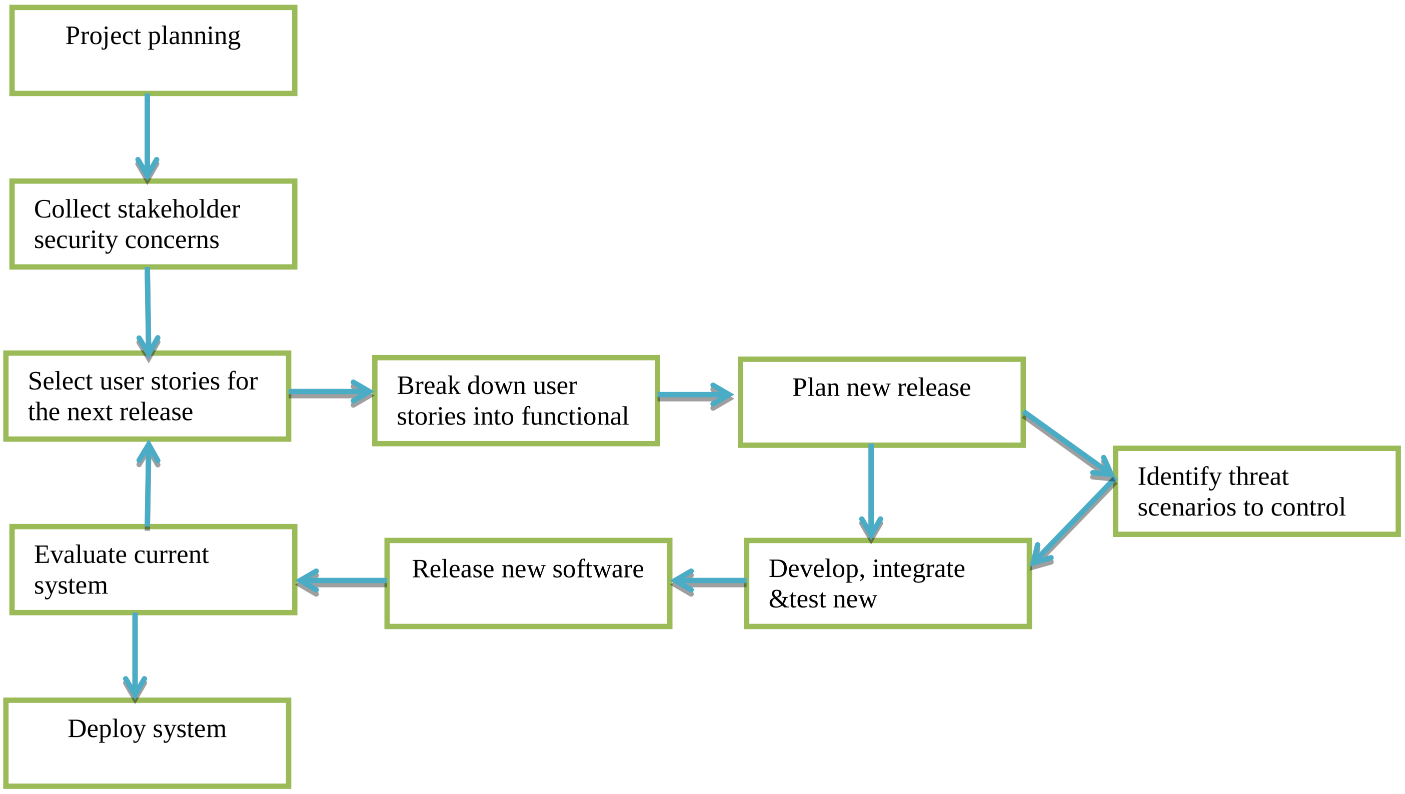
❖ ***Attack models:***
Detention information used to consider similar an attacker: threat observing, abuse condition growth and alteration, data association, and technology-specific attack patterns.

❖ ***Security features and design:***
Preparation is charged with making practical security designs for main security boards,structure middleware

outlines for those panels, and generating and publishing other proactive security guidance.

❖ ***Standards and requirements:***

Training contains producing clear security necessities from the association, determining, structure values for main security panels such as verification, input authentication, and so on, making security values for skills in use, and making a values analysis board.



Project planning
Collect stakeholder security concerns
Select user stories for the next release
Break down user stories into functional
Plan new release
Identify threat scenarios to control
Develop, integrate &test new
Release new software
Evaluate current system
Deploy system

Fig: 3 Proposed models for secure agile software development

3 SDL Touch points:

The SDLTrace opinions area is perhaps the greatest acquainted of the four. This field contains essential software security best practices that remain united into the SDLC. Practices related with examination and pledge of specific software expansion artifacts and procedures. All software security procedures contain these practices. The dual utmost significant software security performs are architecture analysis and code review.

❖ ***Architecture analysis :***

Include taking software architecture in brief diagrams, applying lists of dangers and threats, accepting a procedure for valuation, and building an estimate and remediation idea for the group.

❖ ***Code review:***

Practice contains usage of code review tools, growth of modified rules, profiles for tool usage by dissimilar roles for sample, developers versus analysts, manual analysis, and tracking/measuring results.

4 DEPLOYMENTS:

Perform that interface by traditional network security and software preservation managements. Software conformation, preservation, and other environment problems must straight influence on software security.

❖ ***Penetration testing***

Penetration tests container be spontaneous by software requests or they can be attained manually.

Anyway, the process comprises meeting information about the board previously the test, categorizing believable entry points, trying to interruption and reporting back the answers. The chief objective of penetration testing is to control security weaknesses. A penetration test can likewise be used to test an organization's security plan agreement, its employees' security consciousness and the government's capability to detect and respond to security instances.

❖ ***Software environment***

Practice concerns them self with OS and phase fixing, Web suggestion firewalls, connection and configuration authorization, request watching, change management, and ultimately code signing.

III SECURITY FEATURES:

Security is a developing property of a submission, not just an amalgam of security features. In a try to shorten the problem and make initial strides, organizations often become stuck in a feature-centric mode they tell themselves, “If we just encode our HTTP links and validate users, we’re doing enough.” Thoughtful about how to influence the security features of toolkits, languages, and application servers within an application is good and essential it just isn’t enough. To be successful, the philosophy of “fighting attack” necessity suffuses both and every ESSF activity.

Avoid the feature trap by founding a goal of improving attack fighting in all activity from its beginning.

Some guides:

- ❖ Exercise using vulnerability case educations
- ❖ Describe created on risk, vulnerabilities
- ❖ Avoid conveying on security features check lists
- ❖ Avoid communicating merely on API leader security standers on exercise.

V THE INVESTIGATION PHASE

The upcoming riskanalysis method accepted accessible finished the investigation stage proceeds the technique of a stage to stage process. It determination in to find the information assets that connected that fears vulnerabilities and rank them discussingto those possessions important the extreme defense. Different industries and dissimilar system have adaptable information production requirements.

5.1 Information assets identification and valuation

The listing of assets based on checklist and judgments, yields and adequate identification of the main possessions associated with the software application being developed. These information assets can contain individual information employee salary information customer contact information or economic information. The next step in process is to assign values to each of the key information assets identified this is necessary to determine the impact value and sensitivity of the information in use.

5.2 Threat identification and assessment

It is essential to achieveidentification and valuation threats during the investigation phase of security growth life cycle.

This information is needed to find risks and to guide following designs, coding testing decisions. Such a checklist of the maximum possible threats is helpful in execution a threat valuation, while the software developers need be aware that threat must be continuously changing.

5.3 Risk identification

Risk identification needs that most critical asset and thread connection are recognized to determine which dangers are greatest probable to impact the proposed system. This is complete by just considering the key information assets as recognizedby information assets identification and estimate, and the most likely thread identified that thread identificationand assessment. Those assets with high or dangerous asset impact value risk identification and determine the level of vulnerability.

5.4 Determine the level of vulnerability

In practice security is not cooperated by breaking the dedicated security devices but the exploiting the weakness or vulnerabilities in the method they are used. So the portion of the risk examination procedure it is main to be able to control the level of vulnerability for all risk. The three main levels of vulnerability provided by this model are low, average and high. The next step the risk analysis procedure need that a risk assessment procedure be accepted out to define the extent of each risk.

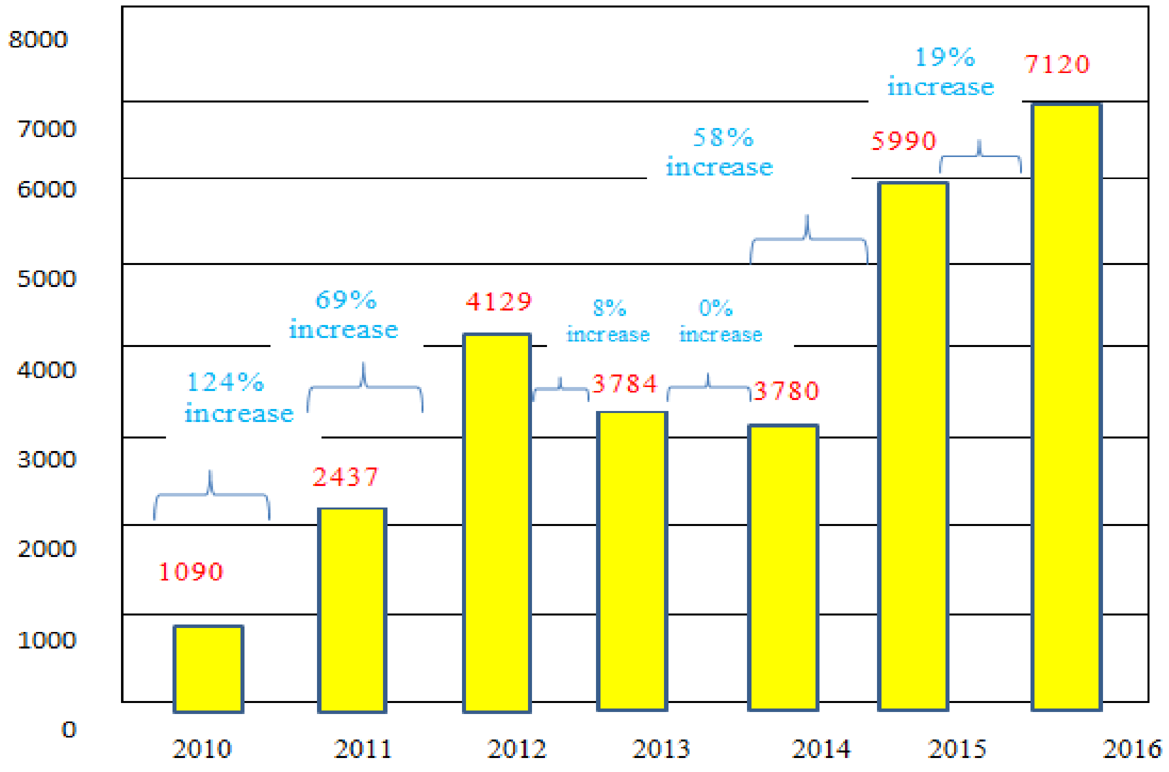
5.5 Risk prioritization

The prioritization risk through the investigation stage serves as guidelines for a analysis design and implementation phase of the security growth life cycle. This is attained by just listing all risk identified risk identification and its consistent risk standards as established at risk valuation.

VI PERFORMANCE ANALYSIS:

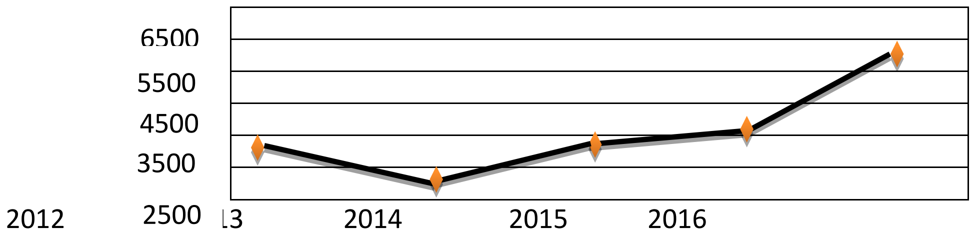
This week	Last week	Weeks on list	Risk	Risk resolution progress
1	1	5	Feature creep	Staged delivery approach adopted nee training
2	-	1	Change of CM system	Evaluation under way
3	5	5	Optimistic schedule	New estimation and functionality prioritizat under way
4	2	5	Program schedule	Negotiations about additional resources under way
5	7	5	Slow customer feedback	Meeting with customer scheduled

Table: 1 Risk and solution analysis



Graph: 1 Vulnerabilities Report

Year	**Number of vulnerabilities**
2012	4258
2013	3532
2014	4347
2015	4794
2016	7038



GRAPH 2 Vulnerabilities of 2012-2016

VII CONCLUSION

Secure software growth demands a countless deal of efforts for of the multistage stages of the software growth life cycle. The secure software development procedure is twofold in nature. On one pointer we still lack the secure software development mechanism that can be extensively accepted and on the other pointer a mechanism is wanted to calculate the efficiency of the applied mechanism. The difficulty behind the immaturity of secure software development procedure is due to the fact that multiple external issues and the variation in the actual operating environment.

After our survey we found that maximum of the efforts towards secure software growth are prepared at the design and testing stages, which is very greatly on the track then design act as a blue print of the whole system. Correspondingly the survey exposed that a great percentage of efforts complete are not appropriate in the actual performs. Our future efforts will be the identification of key factors and parameters answerable for the security of complete system at every of the above mentioned phase of secure software development life cycle. Built on the indicators of the current survey and the identified parameters we will suggest a hybrid secure software development cycle that can be general in nature and appropriate in any environment.

There is an agreement in the industry that security necessity be part of the software developmentlifecycle. It is not a question of which growth model is used, but how wellthe organization is able to integrate security in the procedure.A weak spot in all the models is that they all depend on the team members havingadequate security talents. It cannot be probable that each group must deliver securitytraining to their individual staff. Security training necessity therefore be part of IT education programs in higher education.

There is an urgent essential to strengthen IT education programsworldwide with respect to cyber security. For this purpose it would be interestingto describe a security education maturity model for the university sector. Ifa university proposalsan IT education program with unsatisfactory security focus, then that university is portionof the problem of producing cyber security vulnerabilities. It is time for all IT educationorganizations to developed part of the solutions.

VIII REFERENCES:

1) Wang, Huaiqing, And Chen Wang. Taxonomy of security considerations and software quality. Communications of the ACM 46.6 (2003): 75-78.

2) Devanbu, Premkumar T., And Stuart Stubblebine. Software engineering for security: a roadmap. Proceedings of the conference on the future of Software engineering. ACM, 2000.

3) C. Mann, “Why Software is so bad” Technology Review (July/August 2002)

4) Shirazi H. M., A New Model for Secure Software Development. International Journal of Intelligent Information Technology Application, 2009, 2(3):136-143

5) Boehm, Barry W. Industrial software metrics top 10 list. IEEE software 4.5 (1987): 84-85.

6) Khan, Muhammad Umair Ahmed, And Mohammad Zulkernine. A Survey on Requirements and Design Methods for Secure Software Development. No. 2009-562. Technical Report, 2009.

7) Khan, Muhammad Umair Ahmed, And Mohammad Zulkernine. Activity and Artifact Views of a Secure Software Development Process. Computational Science and Engineering, 2009. CSE'09. International Conference on. Vol. 3. IEEE. 2009.

8) Mir, Irshad Ahmad, and S. M. K. Quadri. "Analysis and evaluating security of component-based software development: A security metrics framework." International Journal of Computer Network and Information Security (IJCNIS) 4.11 (2012): 21.

9) Boehm, Barry W., And Philip N. Papaccio. Understanding and controlling software costs. Software Engineering, IEEE Transactions on 14.10 (1988): 1462-1477.

10) Firesmith, Donald. Specifying reusable security requirements. Journal of Object Technology 3.1 (2004): 61-75.

11) McGraw, Gary. Software Security: Building Security In. Boston,MA: Addison-Wesley, 2006.

12) Nicole Perlroth. A Tough Corporate Job Asks One Question: Can You Hack It? *New YorkTimes Online*, 20 July 2014.

13) Stephen J. Ross. Whiz Bang 2000. *ISACA Journal*, 6, 2014.

14) Dave Wichers. Breaking the Waterfall Mindset of the Security Industry.In *OWASP AppSecUSA*, New York, 2008.

15) Jürjens, Jan. Secure systems development with UML. Springer, 2004.

16) Lodderstedt, Torsten, David Basin, And Jürgen Doser. Secure UML: A UML- based modeling language for model-driven security. «UML» 2002—The Unified Modeling Language (2002): 426-441. 17) FIRESMITH, DONALD G. Security use cases. Journal of object technology 2.3 (2003).

18) Sindre, Guttorm, And Andreas L. Opdahl. Eliciting security requirements with misuse cases. Requirements Engineering 10.1 (2005): 34-44.

19) Mcdermott, John, And Chris Fox. Using abuse case models for security requirements analysis. Computer

Security Applications Conference, 1999.(ACSAC'99) Proceedings. 15th Annual.IEEE, 1999.
20) Hussein, Mohammed, And Mohammad Zulkernine. Umlintr: a UML profile for specifying intrusions."
Engineering of Computer Based Systems, 2006. ECBS 2006. 13th Annual IEEE International Symposium and Workshop on. IEEE, 2006.
21) Raihan, Mohammad, And Mohammad Zulkernine. AsmLSec: an extension of abstract state machine language for attack scenario specification. Availability, Reliability and Security, 2007.ARES 2007.The Second International Conference on.IEEE, 2007.
22) Doan, Thuong, Et Al. "Mac AndUml For Secure Software Design. Workshop on Formal Methods in Security Engineering: Proceedings of the 2004 ACM workshop on Formal methods in security engineering. Vol. 29.No. 29. 2004.
23) Saltzer, Jerome H., And Michael D. Schroeder. The protection of information in computer systems." Proceedings of the IEEE 63.9 (1975): 1278-1308.
24) Bishop, Matt. Introduction to computer security. Addison-Wesley Professional, 2004.
25) Howard, Michael, And David Leblanc. Writing secure code. Microsoft press, 2009.
26) Peine, Holger. Rules of thumb for developing secure software: Analyzing and consolidating two proposed sets of rules. Availability, Reliability and Security, 2008.ARES 08.Third International Conference on.IEEE, 2008.
27) Bauer, Bernhard, Jörg P. Müller, And James Odell. Agent UML: A formalism for specifying multi agent interaction. Agent-oriented software engineering.Vol. 1957.Springer, Berlin, 2001.
28) Graff, Mark, And Kenneth Van Wyk. Secure coding: principles and practices. O'Reilly Media, Incorporated, 2003.
29) Holzmann, Gerard J. The power of 10: rules for developing safety-critical code. "Computer 39.6 (2006): 95-99.
30) Carnegie Mellon university, copyright © 1995-2009 [modified: February 12, 2009], cert, http://www.cert.org/stats/