6-1-2008

SQUARE-Lite: Case Study on VADSoft Project

Ashwin Gayash

Venkatesh Viswanathan

Deepa Padmanabhan

Software Engineering Institute

SQUARE-Lite: Case Study on VADSoft Project

SQUARE Team
Ashwin Gayash
Venkatesh Viswanathan
Deepa Padmanabhan
Dr. Nancy R. Mead, Faculty Advisor and SQUARE Principal Investigator

June 2008

SPECIAL REPORT
CMU/SEI-2008-SR-017

CERT Program

http://www.sei.cmu.edu

Carnegie Mellon

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work is sponsored by Carnegie Mellon CyLab.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html)

.

Table of Contents

List of Tables

Abstract

This special report is the first by the Carnegie Mellon® Software Engineering Institute to focus on the practical application of the SQUARE-Lite security requirements engineering method. Three case study reports about applying the Security Quality Requirements Engineering (SQUARE) process, from which SQUARE-Lite is derived, were published previously.

In this report, the SQUARE and SQUARE-Lite methods are briefly described, and a student team presents the results of working with a client using SQUARE-Lite to develop security requirements for a financial application.

1 Introduction

In September 2007, Carnegie Mellon University and VAD Corporation came together to pilot the SQUARE-Lite methodology on the VAD Corporation's VADSoft project. The SQUARE team and VAD Corporation staff met on a number of occasions from September 2007 through the end of February 2008. The SQUARE team members are listed as authors of this report. The primary contact point at VAD Corporation was from the Information Security Department. He was frequently joined by members of the VADSoft development team as we worked through the pilot activity. The remainder of this report discusses the results of the SQUARE-Lite pilot and lessons learned from the project.

1.1 VAD CORPORATION AND VADSOFT

VAD Corporation is a privately held, medium-sized commercial organization. The VADSoft project is a financial application. User functionality is determined by the client and user roles and functions that are defined by their security model.

2 SQUARE

The SQUARE methodology [Mead 2005] begins with the requirements engineering team and project stakeholders agreeing on technical definitions that serve as a baseline for all future communication. Next, business and security goals are outlined. Third, artifacts and documentation are created, which are necessary for a full understanding of the relevant system. A structured risk assessment determines the likelihood and impact of possible threats to the system. Following this work, the requirements engineering team determines the best method for eliciting initial security requirements from stakeholders, which is dependent on several factors, including the stakeholders involved, the expertise of the requirements engineering team, and the size and complexity of the project. Once a method has been established, the participants rely on artifacts and risk assessment results to elicit an initial set of security requirements. Two subsequent stages are spent categorizing and prioritizing these requirements for management's use in making tradeoff decisions. Finally, an inspection stage is included to ensure the consistency and accuracy of the security requirements that have been generated.

Table 1 details the steps involved in the SQUARE process [Mead 2008].

Table 1: SQUARE Steps

Step 1: Agree on definitions	
Input	Candidate definitions from IEEE and other standards
Technique	Structured interviews, focus group
Participant	Stakeholders, requirements team
Output	Agreed-to definitions
Step 2: Identify security goals	
Input	Definitions, candidate goals, business drivers, policies and procedures, examples
Technique	Facilitated work session, surveys, interviews
Participant	Stakeholders, requirements engineer
Output	Goals
Step 3: Develop artifacts to support security requirements definition	
Input	Potential artifacts (e.g., scenarios, misuse cases, templates, forms)
Technique	Work session
Participant	Requirements engineer
Output	Needed artifacts: scenarios, misuse cases, models, templates, forms
Step 4: Perform risk assessment	
Input	Misuse cases, scenarios, security
Technique	Risk assessment method, analysis of anticipated risk against organizational risk tolerance, including threat analysis
Participant	Requirements engineer, risk expert, stakeholders
Output	Risk assessment results

Step 5: Select elicitation techniques	
Input	Goals, definitions, candidate techniques, expertise of stakeholders, organizational style, culture, level of security needed, cost/benefit analysis, etc.
Technique	Work session
Participant	Requirements engineer
Output	Selected elicitation techniques
Step 6: Elicit security requirements	
Input	Artifacts, risk assessment results, selected techniques
Technique	Joint Application Development (JAD), interviews, surveys, model-based analysis, checklists, lists of reusable requirements types, document reviews
Participant	Stakeholders facilitated by requirements engineer
Output	Initial cut at security requirements
Step 7: Categorize requirements as to level (system, software, etc.) and whether they are requirements or other kinds of constraints	
Input	Initial requirements, architecture
Technique	Work session using a standard set of categories
Participant	Requirements engineer, other specialists as needed
Output	Categorized requirements
Step 8: Prioritize requirements	
Input	Categorized requirements and risk assessment results
Technique	Prioritization methods such as Triage, Win-Win
Participant	Stakeholders facilitated by requirements engineer
Output	Prioritized requirements
Step 9: Requirements Inspection	
Input	Prioritized requirements, candidate formal inspection technique
Technique	Inspection method such as Fagan, peer reviews
Participant	Inspection team
Output	Initial selected requirements, documentation of decision making process and rationale

2.1 SQUARE-LITE

Over the course of the SQUARE case studies, it became clear that SQUARE required a significant commitment on the part of the organizations that used it. Execution of the full SQUARE process could take up to two to three months, and many organizations were not able to make that time commitment.

Therefore we studied the SQUARE process to see which steps might fit into an existing requirements engineering process. We also wanted to come up with a streamlined approach that could provide at least some benefit to organizations that were not prepared to invest in the full SQUARE process. The case study discussed in this report provided us a more precise picture of the benefits of SQUARE-Lite.

SQUARE-Lite consists of four steps extracted from the SQUARE process, Steps 1, 2, 6, and 8. The steps in the SQUARE-Lite process are summarized in Table 2 [Mead 2008].

Table 2: SQUARE-Lite Steps

	Step	Input	Techniques	Participants	Output
1	Agree on definitions	Candidate definitions from IEEE and other standards	Structured interviews, focus group	Stakeholders, requirements team	Agreed-to definitions
2	Identify security goals	Definitions, candidate goals, business drivers, policies and procedures, examples	Facilitated work session, surveys, interviews	Stakeholders, requirements engineer	Goals
3	Elicit security requirements	Artifacts, risk assessment results, selected techniques	Accelerated Requirements Method (ARM), Joint Application Development (JAD), interviews, surveys, model-based analysis, checklists, lists of reusable requirements types, document reviews	Stakeholders facilitated by requirements engineer	Initial cut at security requirements
4	Prioritize requirements	Categorized requirements and risk assessment results	Prioritization methods such as AHP, Triage, Win-Win	Stakeholders facilitated by requirements engineer	Prioritized requirements

2.1.1 Agree on Definitions

To guarantee effective and clear communication throughout the requirements engineering process, the requirements engineers and the project stakeholders must first agree on terms and definitions. The set of terms and definitions chosen by the VADSoft team is shown in the appendix.

2.1.2 Security Goals

The security goals were elicited from the VADSoft team and documented. The security goals are as follows:

- Implement user authentication and access control.
- Protect private customer information from data leaks.
- Ensure that there is no compromise of financial data or actual checks produced by the system.

2.1.3 Risk Assessment

The purpose of assessing risks is to identify the vulnerabilities and threats that the VADSoft project may face and the likelihood that the threats will materialize as real attacks. Therefore the risks pertaining to the VADSoft project were assessed by the team members and the SQUARE team. This step was carried out in addition to the four SQUARE-Lite steps, as it was critical to assess the risks in the project to obtain clarity in the security requirements. A risk matrix and prioritized lists of risks were the outcomes targeted. To facilitate the VADSoft team in eliciting and prioritizing risks, the SQUARE team provided them with a list of risks that the project may face based on the VADSoft requirements documents and inputs from meetings. The VADSoft team then could update this list with other risks that they anticipated. The risk list is shown in Table 3.

Table 3: Risk List

Risk Category	Risk ID	Risk	Probability of Occurrence	Impact
			1-Low, 2-Medium, 3-High	
High-Level	R1	Risk of losing integrity and security of data		
	R2	Risk of breaching security of networks in the organization		
	R3	Risk of data loss in case of some disaster		
	R4	Risk of service loss in case of some disaster		
	R5	Risk of adverse functioning of VADSoft software in case of some threat action		
	R6	Risk of a copy of a check generated by the tool being stored in someone's machine and reprinted		
	R7	Risk of information leak or exposure due to local storage of security-critical data like SSN to improve performance		
	R8	Risk of access to passwords that are not hashed and stored outside the database		
	R9	Risk of developers getting access to the system due to their ability to create roles through the application		
Low-Level (risks due to malware)	R10	Risk of inflicting data loss/corruption due to virus/Trojan attack		
	R11	Risk of stealing confidential information due to virus/Trojan attack		
	R12	Risk of user information gathered through the Internet by spyware		
	R13	Risk of worms exploiting the network, ultimately causing system failure		
Low-Level (risks due to intruders, unauthorized users, or hackers)	R14	Risk of intruder (not authenticated) gaining access to VADSoft software		
	R15	Risk of unauthorized users entering VADSoft software		
	R16	Risk of exposing credit details or tracking numbers to unauthorized users or intruders		
	R17	Risk of deliberate attempt by employees to crack VADSoft software set up within the organization		
Low-Level (database related)	R18	Since VADSoft is a database driven application, risk of database being exposed to threat through SQL injection		

2.2 RISK MATRIX

A risk matrix [ioMosaic 2002] was also prepared and provided by the SQUARE team to the VADSoft team to help them assess the risk exposure of the project. The matrix is filled with the risk IDs from the risk list based on the impact of the risk on the project and on the probability of occurrence. The risk matrix is shown in Table 4.

Table 4: Risk Matrix

1	**< Risk IDs >**	**< Risk IDs >**	**< Risk IDs >**
2	**< Risk IDs >**	**< Risk IDs >**	**< Risk IDs >**
3	**< Risk IDs >**	**< Risk IDs >**	**< Risk IDs >**
Impact	**1**	**2**	**3**
	Probability of Occurrence - Score		

Low Risk Exposure

Medium Risk Exposure

High Risk Exposure

2.2.1 Risk Rank List

Risk rank can be determined from the risk matrix and filled in on the risk rank list. This results in a list of prioritized risks.

Risk ID	Risk Exposure (High/Low/Medium)	Rank (in Numbers)

2.3 ELICIT SECURITY REQUIREMENTS

Security requirements were elicited from the VADSoft team during several elicitation sessions held at VAD Corporation. Certain requirements were taken from the business requirements document for the VADSoft project. The list of security requirements collected is as follows:

- Access to different functionalities of the software should be limited and based on authorization.
- The authorization process should be automatic to improve the usability of the software and should tie to the active directory to ensure that impersonations are not possible. Also with this approach, compromise of user names and passwords can be well managed
- Automate security authorizations for usability purposes.
- Ensure that the right people are given the right access.
- Ensure that only the super user of the system has the right to assign roles and permissions. No other role should have the authorization or the ability to assign permissions and functionality to roles.

- The ability of the customer to view data should be specific to his profile as mentioned in the Customer Maintenance Screen.
- Each customer in the system and his ability to use the system is tailored to his profile. Every customer's profile should be maintained in detail.
- All payment checks issued should be verified.
- Positive pay functionality should be tested on a daily basis to catch fraudulent customer-issued checks.
- Checks should be printed only once. Ensure that checks cannot be saved locally on any machine.
- Securely store cached information.
- Store passwords securely in the system.

2.4 PRIORITIZE SECURITY REQUIREMENTS

The VADSoft team had to prioritize the security requirements identified in the previous step. A traceability matrix that maps security requirements to test cases and risks was also prepared.

2.4.1 Traceability Matrix

The traceability matrix maps the security requirements to test cases and risks. This is to ensure that all the security requirements are addressed by the VADSoft team.

Test Case	Security Requirements	Risk (Risk ID)	Business Requirement Security
Access to the functionality of the application as determined by log-in roles	Access to different functionalities of the software should be limited and based on authorization.	A. Risk of unauthorized users entering VADSoft software (R 15). B. Risk of exposing credit details or tracking numbers to unauthorized users or intruders (R16).	Based on user's user ID and associated role; user will have appropriate level of access to the software and associated data.
Login permission will be tied to a user's Active Directory account. "Remember me" functionality will be incorporated for users associated with one company via a checkbox.	The authorization process should be automatic to improve better usability of the software and should tie to the active directory to ensure that impersonations are not possible. Also with this approach, compromise of user names and password can be well managed.	A. Risk of unauthorized users entering VADSoft (R 15).	Based on user's user ID and associated role; user will have appropriate level of access to the software and associated data.
Ability to remember a user's password via a checkbox.	Automate security authorizations for usability purposes.	A. Risk of unauthorized users entering VADSoft (R 15). B. Risk of exposing credit details or tracking numbers to unauthorized users or intruders (R16).	Based on user's user ID and associated role; user will have appropriate level of access to the software and associated data.

Roles and Permissions Determined by Login	Ensure that the right people are given the right access.	A. Risk of unauthorized users entering VADSoft (R 15). B. Risk of exposing credit details or tracking numbers to unauthorized users or intruders (R16). C. Risk of intruder (not authenticated) gaining access to VADSoft software (R14).	User will be authenticated by network and application when entering VADSoft Based on user's user ID and associated role; user will have appropriate level of access to the software and associated data.
Roles and associated functionality will be determined by the director role based on eight canned templates or completely customized	The director is the super user of the system, and he alone has the right to assign roles and permissions. No other role should have the authorization or the ability to assign permissions and functionality to roles.		Based on user's user ID and associated role; user will have appropriate level of access to the software and associated data.
Some functionality of the VADSoft application will be customer specific. The customer is associated to the login. Behavior of the application will be tailored to the customer profile, specified in the Customer Maintenance Screen and will be unique for each customer	The ability of the customer to view data should be specific to his profile as mentioned in the Customer Maintenance Screen.	A. Risk of unauthorized users entering VADSoft (R 15). B. Risk of exposing credit details or tracking numbers to unauthorized users or intruders (R16).	
Customer Maintenance Screen and will be unique for each customer	Each customer in the system and his ability to use the system is tailored to his profile. Every customer's profile should be maintained in detail.		
Positive pay is an anti-fraud component that utilizes an FTP server to send and verify a list of checks cut that day.	All payment checks issued should be verified.		
Positive pay will be tested by sending a list of checks cut on that day, via FTP, to the bank with which we are doing business. A return message verifying the content will be received. This should be tested for both positive and negative results (i.e., the list matches or the list does not match.)	Positive pay functionality should be tested on a daily basis to catch fraudulent customer-issued checks.		
	Checks should be printed only once. Ensure that checks cannot be saved locally on any machine.	Risk of a copy of a check generated by the tool being stored in someone's machine and reprinted (R6).	
	Securely store cached information.	Risk of information leak or exposure due to local storage of security-critical data like SSN to improve performance (R7).	

	Store passwords securely in the system.	Risk of access to passwords that are not hashed and stored outside the database (R8).	
		Risk of developers getting access to the system due to their ability to create roles through the application (R9).	

2.5 REFLECTIONS

1. The SQUARE team saw a need for a defined process to appropriately incorporate the steps of the SQUARE-Lite methodology.
2. SQUARE fits best when the project is in the initial phases of the software development life cycle. This is because applying SQUARE-Lite results in the identification of prioritized security risks that should influence the development of the software. But in this collaboration with VAD Corporation, the VADSoft project was already in the production phase. The value of SQUARE-Lite's outcome was not realized to its fullest during this phase of development; however, the prioritized requirements list was useful for verifying whether their security requirements were addressed and what further consideration the team should give for the improvement of the product's security.

SQUARE-Lite contains four major steps extracted from the full SQUARE methodology. While applying SQUARE-Lite in the VADSoft project, an additional step for identifying risks was necessary to help identify the security requirements. Hence SQUARE-Lite was slightly modified to incorporate a risk assessment step as defined in SQUARE. The steps in this modified SQUARE-Lite process were

- Agree on definitions
- Identify security goals
- Perform risk assessment
- Elicit security requirements
- Prioritize requirements

Applying SQUARE-Lite required commitment from both the SQUARE team and the VADSoft project team. Steps like identifying risks and prioritizing the requirements depended on input from the members of the VADSoft project team. Since VADSoft was in the production phase and the members were busy in development, there was a delay in completion of the SQUARE-Lite process. This throws light on the fact that cooperation from both ends is necessary for successful and timely delivery of the prioritized security requirements list.

Appendix

The set of terms and definitions chosen by the VADSoft team is shown in Table 5. Definitions after No. 37 were optional.

Table 5: Terms and Definitions

No.	Terms	Definition
1	access control	Access control ensures that resources are only granted to those users who are entitled to them [SANS 2003].
2	access control list	A list of access control entries apply to an entire object, a set of the object's properties or an individual property of an object, and that define the access granted to one or more security principals [Tulloch 2003].
3	artifact	The remnants of an intruder attack or incident activity. These could be software used by intruder(s), a collection of tools, malicious code, logs, files, output from tools, status of a system after an attack or intrusion [West-Brown 2003].
4	attack	An action conducted by an adversary, the attacker, on a potential victim. A set of events that an observer believes to have information assurance consequences on some entity, the target of the attack [Ellison 2003].
5	auditing	The information gathering and analysis of assets to ensure such things as policy compliance and security from vulnerabilities [SANS 2003].
6	authentication	The process of determining whether someone or something is, in fact, who or what it is declared to be [SearchSecurity 2008].
7	authorization	The process of granting a person, computer process, or device access to certain information, services, or functionality. Authorization is derived from the identity of the person, computer process, or device requesting access, which is verified through authentication [Tulloch 2003].
8	availability	The property of a system or a system resource that ensures it is accessible and usable upon demand by an authorized system user. Availability is one of the core characteristics of a secure system [Tulloch 2003].
9	back door	A hardware or software-based hidden entrance to a computer system that can be used to bypass the system's security policies [Tulloch 2003].
10	breach	Any intentional event where an intruder gains access that compromises the confidentiality, integrity, or availability of computers, networks, or the data residing on them [CERT/CC 2004].
11	brute force	A cryptanalysis technique or other kind of attack method involving an exhaustive procedure that tries all possibilities, one by one [SANS 2003].
12	cache poisoning	Malicious or misleading data from a remote name server is saved [cached] by another name server. Typically used with DNS cache poisoning attacks [SANS 2003].
13	confidentiality	The property that information is not made available or disclosed to unauthorized individuals, entities, or processes (i.e., to any unauthorized system entity) [SANS 2003].
14	control	An action, device, procedure, or technique that removes or reduces vulnerability.
15	corruption	A threat action that undesirably alters system operation by adversely modifying system functions or data [SANS 2003].
16	cracker	Someone who breaks into someone else's computer system, often on a network, bypasses passwords or licenses in computer programs, or in other ways intentionally breaches computer security [SearchSecurity 2008].

17	denial-of-service (DoS) attack	An attempt by a malicious (or unwitting) user, process, or system to prevent legitimate users from accessing a resource (usually a network service) by exploiting a weakness or design limitation in an information system. Examples of DoS attacks include flooding network connections, filling disk storage, disabling ports, and removing power [Tulloch 2003].
18	disaster recovery plan	A plan that helps a company recover data and restore services after a disaster [Tulloch 2003].
19	disclosure	A component of the notice principle, wherein a company should make available its data handling practices, including notices on how it collects, uses, and shares personally identifiable information [Tulloch 2003].
20	disgruntled employee	A person in an organization who deliberately abuses or misuses computer systems and their information [Alberts 2003].
21	encryption	The cryptographic transformation of data (called “plaintext”) into a form (called “cipher text”) that conceals the data’s original meaning to prevent it from being known or used [SANS 2003].
22	fault tolerance	A computer system or component designed so that, in the event that a component fails, a backup component or procedure can immediately take its place with no loss of service. Fault tolerance can be provided with software, embedded in hardware, or provided by some combination [SearchSecurity 2008].
23	firewall	A security solution that segregates one portion of a network from another portion, allowing only authorized network traffic to pass through according to traffic-filtering rules [Tulloch 2003].
24	hacker	Someone who engages in the activity of hacking computer programs, systems, or networks [Tulloch 2003].
25	integrity	For data, the property that data has not been changed, destroyed, or lost in an unauthorized or accidental manner [Allen 1999].
26	intrusion	An attempt to compromise a system or network [Tulloch 2003].
27	intrusion detection system	A combination of hardware and software that monitors and collects system and network information and analyzes it to determine if an attack or an intrusion has occurred. Some ID systems can automatically respond to an intrusion [Allen 1999].
28	malware	Programming or files that are developed for the purpose of doing harm. Thus, malware includes computer viruses, worms, and Trojan horses [Webopedia 2008].
29	man-in-the-middle attack	A computer attack during which the cyber criminal funnels communication between a consumer and a legitimate organization through a fake website. In these attacks, neither the consumer nor the organization is aware that the communication is being illegally monitored. The criminal is, in effect, in the middle of a transaction between the consumer and his or her bank, credit card company, or retailer [BSA 2008].
30	non-repudiation	A technique used to ensure that someone performing an action on a computer cannot falsely deny that they performed that action [Tulloch 2003].
31	patching	The process of updating software to a new version that fixes bugs in a previous version [SANS 2003].
32	recovery	A system’s ability to restore services after an intrusion has occurred. Recovery also contributes to a system’s ability to maintain essential services during intrusion [Ellison 2003].
33	risk	The product of the level of threat with the level of vulnerability. It establishes the likelihood of a successful attack [SANS 2003].
34	threat	A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm [SANS 2003].
35	trust	Determines which permissions other systems or users have and what actions they can perform on remote machines [SANS 2003].
36	vulnerability	Anything that gives an attacker the opportunity to perform an exploit.
37	liability	The responsibility of someone for damage or loss [West-Brown 2003].

38	luring attack	A type of elevation of privilege attack where the attacker “lures” a more highly privileged component to do something on his or her behalf. The most straightforward technique is to convince the target to run the attacker’s code in a more privileged security context [Brown 2005].
39	mail relaying	A practice in which an attacker sends email messages from another system’s email server in order to use its resources and/or make it appear that the messages originated from the other system.
40	malicious code	Software that fulfills the deliberately harmful intent of an attacker when run. For example, viruses, worms, and Trojan horses are malicious code.
41	malicious user	A user who accesses a system with the intent to cause harm to the system or to use it in an unauthorized manner.
42	masquerade	Attempts to fool other machines on the network into accepting the imposter as an original, either to lure the other machines into sending it data or to allow it to alter data [Howard 1998].
43	modification	Situation in which an unauthorized party not only gains access to but tampers with an asset [Howard 1997].
44	non-essential services	Services to users of a system that can be temporarily suspended to permit delivery of essential services while the system is dealing with intrusions and compromises [Ellison 1997].
45	patch	A small update released by a software manufacturer to fix bugs in an existing program [SANS 2003].
46	penetration	Intrusion, trespassing, or unauthorized entry into a system [RUsecure 2008].
47	penetration testing	The execution of a testing plan, the sole purpose of which is to attempt to hack into a system using known tools and techniques [RUsecure 2008].
48	permissions	Authorization to perform operations associated with a specific shared resource, such as a file, directory, or printer. Permissions must be granted by the system administrator to individual user accounts or administrative groups.
49	physical security	Security measures taken to protect systems, buildings, and related supporting infrastructure against threats associated with their physical environment [Guttman 1995].
50	port scanning	The act of systematically scanning a computer’s ports [Webopedia 2008].
51	privacy	The quality or condition of being secluded from the presence or view of others [Dictionary.com 2008].
52	privacy policy	An organization’s requirements for complying with privacy regulations and directives. The policy is expressed in a privacy statement.
53	procedure	The implementation of a policy in the form of workflows, orders, or mechanisms [West-Brown 2003].
54	recognition	The capability of a system to recognize attacks or the probing that precedes attacks [Ellison 2003].
55	replay attack	The interception of communications, such as an authentication communication, and subsequent impersonation of the sender by retransmitting the intercepted communication [FFIEC 2008].
56	request for collaboration (RFC)	A request for development engagement where Product Support Services (PSS) is technically blocked; also used to formalize and track support statement requests and to review proposed action plans. The request process was introduced to help reduce the time it takes to provide a solution to a customer. An RFC may become a DCR [design change request], CDCR [critical design change request], or hotfix request [a request to address a problem in a product] [Microsoft 2005].
57	resilience	The ability of a computer or system to both withstand a range of load fluctuations and remain stable under continuous and/or adverse conditions [RUsecure 2008].
58	resistance	The capability of a system to resist attacks [Ellison 2003].
59	risk assessment	The process by which risks are identified and the impact of those risks determined [SANS 2003].

60	script kiddie	The more immature but unfortunately often dangerous exploiter of security lapses on the Internet. The typical script kiddie uses existing and frequently well-known and easy-to-find techniques and programs or scripts to search for and exploit weaknesses in other computers on the Internet—often randomly and with little regard or perhaps even understanding of the potentially harmful consequences [SearchSecurity 2008].
61	security policy	A policy that addresses security issues [West-Brown 2003].
62	SQL injection	A type of input validation attack specific to database-driven applications where SQL code is inserted into application queries to manipulate the database [SANS 2003].
63	spoof	Making a transmission appear to come from a user other than the user who performed the action [Microsoft 2005].
64	stakeholder	Anyone who is a direct user, indirect user, manager of users, senior manager, operations staff member, support (help desk) staff member, developer working on other systems that integrate or interact with the one under development, or maintenance professionals potentially affected by the development and/or deployment of a software project [Ambler 2004].
65	stealthing	A term that refers to approaches used by malicious code to conceal its presence on an infected system [SANS 2003].
66	survivability	The capability of a system to complete its mission in a timely manner, even if significant portions are compromised by attack or accident. The system should provide essential services in the presence of successful intrusion and recover compromised services in a timely manner after intrusion occurs [Mead 2003].
67	target	The object of an attack, especially host, computer, network, system, site, person, organization, nation, company, government, or other group [Allen 1999].
68	threat assessment	The identification of the types of threats that an organization might be exposed to [SANS 2003].
69	threat model	Used to describe a given threat and the harm it could to do a system if it has a vulnerability [SANS 2003].
70	toolkits	A collection of tools with related purposes or functions, e.g., antivirus toolkit, disk toolkit [RUsecure].
71	Trojan	A program that appears to be useful or harmless but that contains hidden code designed to exploit or damage the system on which it is run. Trojan horse programs are most commonly delivered to users through e-mail messages that misrepresent the program's purpose and function. Also called Trojan code [Microsoft 2005].
72	upgrade	A software package that replaces an installed version with a newer version of the same software. The upgrade process typically leaves existing customer data and preferences intact while replacing the existing software with the newer version.
73	user profile	Settings that define customization preferences for a particular user, such as desktop settings, persistent network connections, personally identifiable information, website use, or other behaviors and demographics data.
74	user rights	Tasks that a user is permitted to perform on a Windows-based computer or domain. There are two types of user rights: privileges and logon rights. An example of a privilege is the right to shut down the system. An example of a logon right is the right to log on to a computer interactively. Both types are assigned by administrators to individual users or groups as part of the security settings for the computer.
75	victim	That which is the target of an attack. An entity may be a victim of either a successful or unsuccessful attack [SANS 2003].
76	virus	A hidden, self-replicating section of computer software, usually malicious logic, that propagates by infecting—i.e., inserting a copy of itself into and becoming part of—another program. A virus cannot run by itself; it requires that its host program be run to make it active [SANS 2003].
77	worm	Self-propagating malicious code that can automatically distribute itself from one computer to another through network connections. A worm can take harmful action, such as consuming network or local system resources, possibly causing a denial-of-service attack [Microsoft 2005]. Compare virus.

References

[Alberts 2003]
Alberts, Christopher & Dorofee, Audrey. *OCTAVE Threat Profiles*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003.
http://www.cert.org/archive/pdf/OCTAVEthreatProfiles.pdf

[Allen 1999]
Allen, J., Christie, A., Fithen, W., McHugh, J., Pickel, J., & Stoner, E. *State of the Practice of Intrusion Detection Technologies* (CMU/SEI-99-TR-028, ADA375846). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999. http://www.sei.cmu.edu/publications/

[Ambler 2004]
Ambler, Scott W. *Active Stakeholder Participation*. Evergreen, Colorado, Ronin International, Inc., 2004. http://www.agilemodeling.com/essays/activeStakeholderParticipation.htm

[Brown 2005]
Brown, Keith. "Item 7: What is a Luring Attack?" in *The .NET Developer's Guide to Windows Security*. Boston, MA: Addison-Wesley, 2005.

[BSA 2008]
Business Software Alliance. "Man-in-the-Middle Attack." *Cyber Safety Glossary*, 2008.
http://www.bsacybersafety.com/threat/man-in-the-middle.cfm

[Dictionary.com 2008]
Dictionary.com. Los Angeles, CA: Lexico Publishing Group, LLC.
http://dictionary.reference.com/

[Ellison 1997]
Ellison, B., Fisher, D. A., Linger, R. C., Lipson, H. F., Longstaff, T., & Mead, N. R. *Survivable Network Systems: An Emerging Discipline* (CMU/SEI-97-TR-013, ADA341963). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1997.
http://www.sei.cmu.edu/publications/

[Ellison 2003]
Ellison, R. & Moore, A. *Trustworthy Refinement Through Intrusion-Aware Design* (CMU/SEI-2003-TR-002, ADA414865). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu/publications/

[FFIEC 2008]
FFIEC. "FFIEC Information Technology Examination Handbook Glossary." Washington, D.C.: Federal Financial Institutions Examination Council.
http://www.ffiec.gov/ffiecinfobase/html_pages/gl_01.html

[Guttman 1995]
Guttman, Barbara & Roback, Edward. *An Introduction to Computer Security: The NIST Handbook*. Gaithersburg, MD: U.S. Department of Commerce, Technology Administration, National Institute of Standards and Technology, 1995. http://csrc.nist.gov/publications/nistpubs/800-12/handbook.pdf

[Howard 1997]
Howard, John D. *An Analysis of Security Incidents on the Internet 1989-1995*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1997. http://www.cert.org/research/JHThesis/Start.html

[Howard 1998]
Howard, John & Longstaff, Thomas. *A Common Language for Computer Security Incidents*. Albuquerque, NM: Sandia National Laboratories, 1998. http://www.cert.org/research/taxonomy_988667.pdf

[ioMosaic 2002]
ioMosaic Corp. "Designing an Effective Risk Matrix." An ioMosaic Corporation Whitepaper, 2002. http://archives1.iomosaic.com/whitepapers/risk-ranking.pdf

[Mead 2003]
Mead, Nancy. *Requirements Engineering for Survivable Systems* (CMU/SEI-2003-TN-013, ADA418410). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu/publications/

[Mead 2005]
Mead, N. R., Hough, E. & Stehney, T. *Security Quality Requirements Engineering (SQUARE) Methodology* (CMU/SEI-2005-TR-009). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005. http://www.sei.cmu.edu/publications/

[Mead 2008]
Mead, N. R. "SQUARE: Requirements Engineering for Improved System Security." http://www.cert.org/sse/square.html (2008).

[Microsoft 2005]
Microsoft Corp. Microsoft Security Glossary. http://www.microsoft.com/security/glossary.mspx (2005).

[RUsecure 2008]
RUsecure. The Information Security Glossary. http://www.yourwindow.to/information-security/

[SANS 2003]
The SANS Institute. "SANS Glossary of Terms Used in Security and Intrusion Detection." http://www.sans.org/resources/glossary.php (2003).

[SearchSecurity 2008]
SearchSecurity.com. http://searchsecurity.techtarget.com/

[Tulloch 2003]
Tulloch, M. *Microsoft Encyclopedia of Security*. Redmond, WA: Microsoft Press, 2003.

[Webopedia 2008]
Webopedia.com. http://www.webopedia.com/

[West-Brown 2003]
West-Brown, M., Stikvoort, D., Kossakowski, K., Killcrece, G., Ruefle, R., & Zajicek, M. *Handbook for Computer Security Incident Response Teams (CSIRTs)* (CMU/SEI-2003-HB-002). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003.
http://www.sei.cmu.edu/publications/

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. **AGENCY USE ONLY** (Leave Blank)	2. **REPORT DATE** June 2008	3. **REPORT TYPE AND DATES COVERED** Final
4. **TITLE AND SUBTITLE** SQUARE-Lite: Case Study on VADSoft Project		5. **FUNDING NUMBERS** FA8721-05-C-0003
6. **AUTHOR(S)** SQUARE Team (Ashwin Gayash, Venkatesh Viswanathan, Deepa Padmanabhan, Dr. Nancy R. Mead, Faculty Advisor and SQUARE Principal Investigator)		
7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213		8. **PERFORMING ORGANIZATION REPORT NUMBER** CMU/SEI-2008-SR-017
9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)** HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116		10. **SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. **SUPPLEMENTARY NOTES**		
12A **DISTRIBUTION/AVAILABILITY STATEMENT** Unclassified/Unlimited, DTIC, NTIS		12B **DISTRIBUTION CODE**
13. **ABSTRACT (MAXIMUM 200 WORDS)** This special report is the first by the Software Engineering Institute focusing on the practical application of the SQUARE-Lite security requirements engineering method. Three case study reports about application of the Security Quality Requirements Engineering (SQUARE) process, from which SQUARE-Lite is derived, were published previously. In this report, the SQUARE and SQUARE-Lite methods are briefly described, and a student team presents the results of working with a client using SQUARE-Lite to develop security requirements for a financial application.		
14. **SUBJECT TERMS** requirements engineering, requirements elicitation, security requirements, system security, SQUARE, SQUARE-Lite, case study		15. **NUMBER OF PAGES** 22
16. **PRICE CODE**		

17. **SECURITY CLASSIFICATION OF REPORT** Unclassified	18. **SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	19. **SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	20. **LIMITATION OF ABSTRACT** UL