Estimating Security Risk for Web Applications using Security Vectors

Hui Guan[1, 3, *], Wei-Ru Chen[1], Lin Liu[2], and Hong-Ji Yang[3]

[1] School of Computer Science and Technology,
Shenyang University of Chemical Technology,
Shenyang , China
guanh1999@126.com, willc@china.com

[2] School of Software,
Tsinghua University,
Beijing, China
linliu@tsinghua.edu.cn

[3] Software Technology Research Laboratory,
De Montfort University,
Leicester, England
hyang@dmu.ac.uk

Received 18 September 2011; Revised 12 January 2012; Accepted 15 February 2012

Abstract. Risk assessment has been getting increased attention as the new vulnerabilities and threats are emerging on daily basis. The popularity and complexity of web application present challenges to the security implementation for web engineering. It is well known that the earlier to perform risk assessment for software, the less cost needed to mitigate the security risks. However, quantitative estimation of security in the earlier stage of software development life cycle is largely missing. In this paper, we propose a quantitative approach to perform risk assessment at design stage for web application which is based on multiple security vectors of asset, threat and vulnerability. An environment-driven method is proposed to elicit threats to the system. In the end, the risk assessment methodology is applied on a customer goods case study.

Keywords: risk assessment; threat; security; asset; vulnerability; design stage

1 Introduction

The ease of implementation and use of web technologies has made them an omnipresent and essential component of online commercial sites, intranet and extranet application, as well as the internet services offered and used by companies, which creates new challenges for the web engineering. This is because web applications open systems and information to be accessed by public. In this type of software, requirements have become more complex, in order to guarantee information security [1]. Comprised web server can damage organizations in many ways, from surrendering customer privacy data and accepting fraudulent transactions to indirectly damaging corporate reputation as the result of defaced homepage. Report has shown that web applications account for 80 percent of internet vulnerabilities in the second half of 2008 and rose in prevalence by about eight percent from the first half of the year [2]. Security in web engineering has become an emergent task.

Given the increasing complexity and frequency of web application risks, decision makers must take action to sufficiently protect the web applications. However, security as an architectural driver is often at the expense of performance (e.g. component redundancy), usability (e.g. complexity of using the application) and cost (e.g. using SSL to implement HTTP requires PKI or third party certificates, slows traffic, etc.). Most development companies are having a tough time balancing all of these factors [3]. Thus, many organizations bring security to the forefront of web applications design only after an incident occurs. The result is generally an expensive, knee-jerk reaction to security problems that might have been avoided with intelligently planned controls [4]. There is no denying the fact that web application addressed as early as the design stage, during development and integration, and throughout the life cycle of the application. It must be an integrated component of the application and not be added on at the end of the development cycle. It is a far more cost efficient and effective way than applying security features in a haphazard manner.

*Correspondence author

Risk assessment is a process of assessing the security level in a quantitative or qualitative mode of an organization by evaluating one's exposure to the threats to its assets and operating capabilities. Risk assessment for web application is one of the effective methods to help decision makers determining how much they need to invest in security so as to achieve a desired level. Doing this ensures that all risks have been taken into account and makes it possible to find appropriate security solutions and measures based on the likelihood and impact of identified potential risks. Threat risk modeling methods can be used to facilitate this process.

In this paper, we extend our previous work in [25] to a quantitative risk assessment model which can be used in software design stage to anticipate the risk level. A quantitative method is depicted to ease assessing risks for web application by defining web application classifications which is proposed taking consideration of security-related environmental factors. The remainder part of this paper is arranged as follows. Section 2 reviews the related works in this area. Section 3 illustrates our proposed model and its detailed context, while Section 4 elaborates the proposed approach by a case study and results have been discussed. Finally, Section 5 concludes by summarizing the key contributions and outlining the future steps.

2 Related Works

Making risk management an integral part of the software development process allows it to drive the development process so that security issues are ameliorated early in the product's life [29]. Developers are expected to identify, rank, mitigate and manage risk throughout the software product life cycle. Methodologies such as traditional risk assessment approaches, threat and vulnerability identification that are used to allow risk to drive the development process have in large part been qualitative in nature.

We would like to stress that the class/style files and the template should not be manipulated and that the guidelines regarding font sizes and format should be adhered to. This is to ensure that the end product is as homogeneous as possible.

2.1 Risk Assessment

There are methods and tools confirm the state of the art in risk assessment, such as MAGERIT, CRAMM and OCTAVE. British government encourages industry using certification based on BS7799 [26], which is the information security management standard. They regulate that risk analysis should be included in information security management system establishment. Each method and tool has its own benefits. However, they do not address specific aspects apply to web application.

SP 800-30, Fips 65 [30] is the Risk Management Guide for Information Technology Systems developed by the National Institute of Standards and Technology. MAGERIT is an open methodology for Risk Analysis and Management, developed by the Spanish Ministry of Public Administrations, offered as a framework and guide to the Public Administration [31]. OCTAVE [33] is a heavyweight risk methodology approach originating from Carnegie Mellon University's Software Engineering Institute (SEI) in collaboration with CERT. OCTAVE focuses on organizational risk, not technical risk. However, OCTAVE is large and complex, with many worksheets and practices to implement and it does not provide a list of "out of the box" practices for assessing and mitigating web application security risks.

CVSS is a complicated scoring system composed of three metric groups developed by the US Department of Homeland Security (DHS) [32]. The disadvantage of CVSS is that it does not find or reduce the attack surface area or help enumerate risks within the target system.

Except for the risk assessment methods and tools discussed above, there are many methods proposed to figure out the risk assessment for web application [1], [34], [35]. Cock D. et al. [34] used threat modeling for security tokens in web applications. They pointed out all the possible threats of web-applications, but missed the threats related to the environment where the applications were hosted. Brunil et al. [1] proposed a methodological tool for web application focused on one of the risk assessment steps--asset identification. [1] and [35] use STRIDE model to perform the risk assessment for web application but lack of threat identification. [20] propose a flow-based model to identify and classify the threats for web applications.

2.2 Threat Identification

Threat elicitation is within the domain of software security requirement. In our earlier work, we focus on the security design using social modeling concept [22], [23] and a systematic evaluation of security requirements is proposed [24], [25]. A 3-dimensional vector for quantitative evaluation of security requirements has been proposed, which takes into account the importance of assets to be protected, the vulnerability of the system and the

trustworthiness of environment [24]. Social modeling concept is used to analyze the business and organizational context of systems with regard to security [22], [23].

In terms of threat elicitation, several methods have been proposed. One type of proposed method is abuse cases, misuse cases approaches [5]-[8] which can be used once the system use cases have been created. Abuse and misuse cases [5]-[7] are independent use cases initiated by external attackers to the system which can be defined as a sequence of actions, including variants, that a system or other entity can perform, interacting with misusers of the entity and causing harm to some stakeholder if the sequence is allowed to complete [5].Threat can be elicited by analyzing misuse cases. The goal of misuse cases is to decide and document a priori how software should react to illegitimate use. [8] extends the misuse cases to misuse activities which are analyzed to see how it could be subverted to produce a misuse of information, as a result, a set of threats can be listed. However, the practical method for creating misuse cases or activities is usually with the process of brainstorming [5].

The threat modeling approach [9]-[12], [16] is another approach which gives a clear idea of how to elicit, classify, prioritize and mitigate threats. Especially, some threat modeling approaches are designed for web applications [13]-[15]. Tøndel et al [10] shows that threat modeling often is considered as an important part of the requirements phase, as well as an iterative process, continuously revisited throughout the software lifecycle. Oladimeji et al. [16] propose a goal-oriented approach to threat modeling where the notions of negative softgoals are used for representing threats. In [13] a comprehensive approach is provided to building highly secure and feature-rich web applications using the .NET Framework. [14] elaborates, illustrates and discusses the threat modeling process and its usefulness to the architectural designs of an e-banking application. Although threat modeling is seen as a thorough approach to threat elicitation, mitigation and management, however, it demands information available only at late design time which drives the security design to start in the middle of the software development life cycle.

Another method to elicit the threat is problem frame [17], [18]. A problem frame characterizes a class of simple problem. Realistic problems are seen as compositions of simple problems of recognized to elicit and analyze software security requirements [18], [19]. Hatebur et al. [19] describe a security engineering process to develop security systems based on problem frames, and a collection of security patterns, plus components as the way to deal with the solution. While problem frames appear useful in some cases, they are not as useful for a complete design as UML. Also, they are not so widespread.

The above risk assessment has focused on process/method of general risk analysis, but our focus is to assess risk at design stage using security risk vectors. It assists many organizations to apply risk analysis in their early development stage.

3 Proposed Approach

In order to make a precise description of our proposed model, the concepts involved in the proposed risk analysis approach should be identified and defined.

- Asset is “anything that has value to the organization” and which therefore requires protection.
- Threat is the potential cause of an unwanted event (i.e., an attack), which may result in harm of a system or organization.
- Vulnerability is a weakness of an asset or control (i.e., in ISO 27000-series a control is a synonym of a countermeasure), which may be exploited by a threat. This general definition covers all threats categories.
- Risk is the combination of the probability of an event and its consequence.
- Attack is an attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of an asset.

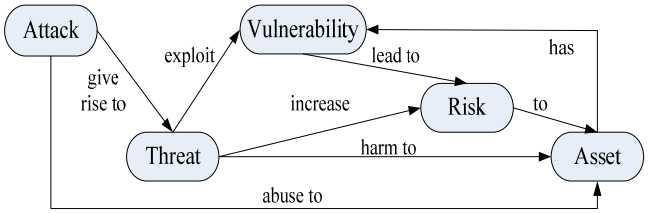

Fig. 1. Security concepts relationship

*Remark 1.*Fig. 1 shows how different security factors, involved in risk analysis, are related. It is obvious that the target of attack is the assets whose vulnerabilities are exploited by threats which in turn lead to risks and do harm to assets.

In this paper, we propose a quantitative risk assessment method for web application taking consideration of criticality of asset, threat and vulnerability.

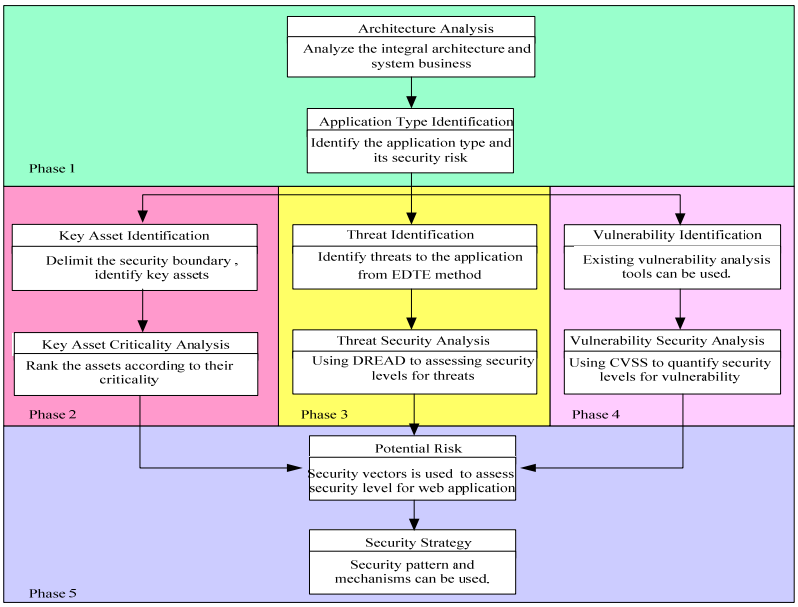

Fig. 2. Proposed risk analysis process

Our propose model consists of five phases with difference phases in different colors. More than one steps may be needed for every phases. Here are the detailed meanings of each phase.

Phase 1: Architecture and Environment Analysis

Step 1: Architecture Analysis. System architecture analysis is the recognition process of entire system architecture and business processes so as to precisely understand the platform structure, security boundary, business processes, internal and external environment of the target system. The system architecture model can be established which is the foundation for data flow analysis.

Step 2: Application Type Identification. In order to have a precise risk estimation of the target web application, it needs to be classified into one of types (from web-app1 to web-app6) according to our web application classification. When it is done, the security risks of this kind of web application can be evaluated preliminarily.

Phase 2: Key Asset Analysis

Step 1: Key Assets Identification. In this step, recognition and analysis of the key information assets that is the key data and services which determine system security should be identified. The key information assets are the kernel for risk assessment.

Step 2: Asset Criticality Ranking. The criticality of assets is determined based on the type of information handled in the applications.

Phase 3: Threat Analysis

Step 1: Threats Identification. Threat identification is the process of recognizing the threats related to each key asset identified in phase 2. The result of threat identification is a list of threats associated with the target system. In this case, we use our EDTE [25] method to elicit threats for web applications.

Step 2: Threat Quantification. Threat quantification is an important step to risk assessment and it is the process of quantifying the threats listed in the previous step by using threat risk assessment model DREAD [36] which is part of a system for classifying computer security threats used at Microsoft. It provides a mnemonic for risk rating security threats using five categories. A risk value may be calculated to each threat after this step.

Phase 4: Vulnerability Analysis

Step 1: Vulnerability Identification. The diagnosis tools of network and host vulnerability can be used to perform vulnerability analysis. A list of system vulnerabilities can be detected after this step.

Step 2: Vulnerability Security Scoring. There are a number of vulnerability “scoring” systems such as Common Vulnerability Scoring System (CVSS)[32]. We can perform CVSS to rate each vulnerability that identified from the previous step, and produce a total score for the vulnerability of system.

Phase 5: Risk and Security Strategy Analysis.

Step 1: Potential Risk will be evaluated based on the results of asset, threat and vulnerability analyses using our security vector <A, T, V> formula (1).

Step 2: Security Strategy. Safeguards and their appropriateness are investigated and analyzed in this process. The validation of the new safeguard applied in the future needs to be also checked.

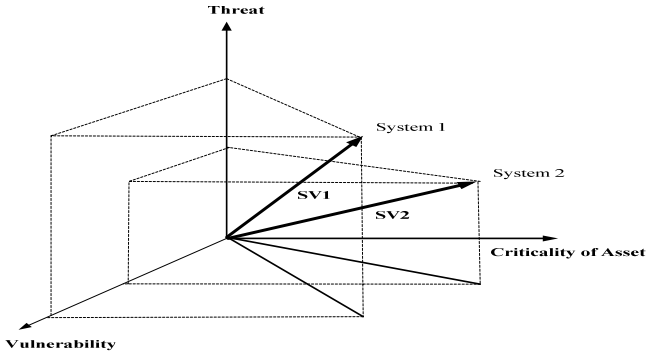

Fig. 3. Security vectors

From Fig.3 we can conclude that the computation of security evaluation vectors is the combination of the factors themselves (A for Asset, T for Threat, V for Vulnerability) and the weight of each factor.

$$SV=\sqrt{A^2\times Wa+T^2\times Wt+V^2\times Wv} \quad \textbf{(1)}$$

Where Wa for the weight of asset, Wt for the weight of threat and Wv for weight of vulnerability. The values of Wa, Wt and Wv range from 1 to 1. In this case, taking consideration of importance of each factors, the weights of them are the same which can be formulized as

$$SV=\sqrt{(A^2+T^2+V^2)/3}\ . \quad \textbf{(2)}$$

3.1 Classification of Web Application

All web-applications are not same, the architecture and its supporting systems could be different for each application depending on the complexity, but the resources or techniques needed for running those applications may be same. So, it is possible to create a common threat model and identify all possible threats that can be used for all web-applications [20]. The Web Application Classification is a cooperative effort to clarify and organize the types of web application with different security risk level so as to ease the process of threats identification. It helps you to identify which threats are relevant to your application through the proposed model.

As has discussed in Section 1, the risks web application faced are certain to be different when they are hosted in well secured and non-secure environments. Thus, it is necessary for us to take consideration of the web application type before further discussion.

For the consideration of the environment where web application hosted, three attributes of web applications have been taken into account, which are the usage scope, target user and connectivity mode. Then, the circumstances of all the attributes of web application are listed in Table 1. Some symbols are used to represent these attributes, US (Internal use only, Internal and External use, External use only) for the usage scope and its values, TU (Known users, anonymous) for target user, and CM (Intranet, VPN, Internet) for connection mode.

Table 1. Attributes of Web Application

Usage Scope (US)	Target User (TU)	Connection Mode (CM)
Internal use only (US1)	Know users (TU1)	Intranet (CM1)
Blended use (US2)	Anonymous (TU2)	VPN (CM2)
External use only (US3)		Internet (CM3)

The number of web application types abbreviated as WA equals to:

$$WA=\text{Card}(\text{Domain}(US))\times\text{Card}(\text{Domain}(TU))\times\text{Card}(\text{Domain}(CM))\ . \quad \textbf{(3)}$$

Where Card(Domain(US)) denotes the cardinality of Domain(US). There should be 18 web application types according to the formula 3. However, some of them are not applicable in real web application which should be ignored. For example, the combinations of (US1, TU1, CM3) or (US1, TU2, CM3) are not incompatible. All of

the applicable web application types are listed in Table 2 with the bigger value the higher possibility of risk level.

Table 2. Web Application Classification

ID	Name	Attribute	Definition	Security risks
WA1	Internal use facing known users via intranet	US1 TU1 CM1	Application used primarily on the internal network of an organization for a mount of known users.	This kind of applications is designed for internal use so that only internal users can access from intranet. Therefore, the security risk is considered as low.
WA2	Internal blended External use facing known users via VPN	US2 TU1 CM2	Application used primarily on the internal network of an organization, but a mount of known external clients can access through VPN	The security risk is low but there are possibilities for sharing user-credentials, impersonation and sniffing on the external client site.
WA3	External use facing known users via Internet	US3 TU1 CM3	Application used for external use. A mount of known users can access from internet	The security risk is a bit higher compared to previous types because it is exposed to all kinds of attacks from internet, however, it is not very high for only known users can access
WA4	External use facing public users via Internet	US3 TU2 CM3	Application used for external use. Public users can access from internet	The security risks of these applications are considered little bit high compared to previous types since they are open to public from Internet
WA5	Internal blended external use facing known users via Internet	US2 TU1 CM3	Application used for internal users and external known users from Internet	The security risks of these applications are higher due to their design complexities. Usually, this kind of applications are designed primarily for internal use, it is a little more dangerous when known users access from Internet
WA6	Internal blended external use facing public users via Internet	US2 TU2 CM3	Application used for internal users and external public users from Internet	The security risks of these applications are highest due to their design complexities. Usually, this kind of applications are designed primarily for internal use, it is the most dangerous when public users access from Internet due to lack of security controls

3.2 Asset Analysis

In this paper, we propose an environment critical asset assessment method to perform the asset identification and asset criticality analysis. The asset assessment process is briefly explained based on the international standard ISO 17799, improved by BS7799 [26]. The method to identify asset used in this paper is similar to [27]. The proposed method to assess information system assets includes analyzing information security requirements, understanding criticality of asset, and checking sensitivity for data asset.

Asset Identification. The British BS7799 [26] suggests the asset classification as follows:

- Information Asset: DB, data file, system document, user manual, study and training materials, regulations for management, plan document, provision for alternative system
- Documents: contracts, guidelines, company documents, important business documents
- Software Asset: applications S/W, system S/W, development tool and utility
- Physical Asset: computer and communication equipment, magnetic tape, magnetic disk, power supply, air conditioner, furniture, facilities
- Personnel Asset: individuals, customer, subscriber
- Image and Reputation of a Company
- Service: computer and communication service, warm, light, air conditioning

This paper doesn't deal with all the listed assets in an organization but assets related only to major information system. Thus, we consider only the following asset domains:

- Software: including applications to support the objectives and business processes in an organization
- Information: including data to achieve the objectives and business process

Asset Security Analysis. We treat the criticality of assets according to their property and the environment where they host. If the application deals with personal identity such as name, password, and finance related information such as credit card and bank account information, and the application is designed for external use, the criticality level is treated as high. Otherwise, the criticality level is treated as medium if the application deals with personal information for internal use, and low if it does not deal with personal information. The relationship of asset criticality with the processed data and its applied environment is shown in Table 3.

Table 3. Asset Criticality Evaluation

Asset type	Personal or sensitive	Environment	Criticality level	Risk rating scale
Data Application	No	Internal use	Very low	0.0-1.99
	No	External use	Low	2.0-3.99
	Yes	Internal use	Medium	4.0-5.99
	Yes	External facing known users	High	6.0-7.99
	Yes	External facing public users	Very high	8.0-10.0

3.3 Threat Identification

The methods of threat elicitation mentioned in Section 2 are general purpose for all kinds of applications. However, web application is some kind of different one due to its application environment and complexity. Accordingly, threats to this kind of software are different to some extent. The number and category of the threats to different kind of web applications may not be the same when taking account of complexity and environment where the applications are hosted. Sometimes, the applications may be developed with security in mind, and may be difficult to penetrate as well, but if the environment where the application is hosted is not properly secured, it is easy to penetrate the environment , and as a result, it is easy to compromise the whole application including its subsystems and platform.

As far as the web application developers are concerned, on one hand, they need to keep security on mind when developing the web application, on the other hand, they are usually forced to face the dilemma that how to trade-off among so many product factors such as security requirements, product deadline and budgets etc. Thus, this paper proposes a novel approach to ease the elicitation of the threats for web applications by defining web application classification as the filter to rule some threats out immediately according to the security requirement and the given scenery. Before diving into the details of the proposed model, it is better to give an overall idea of this model, described in Fig.4.

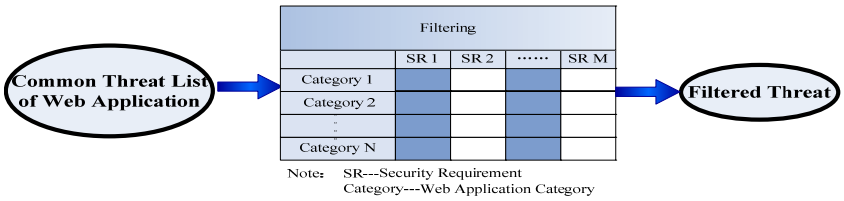

Fig. 4. Illustration of the Environment-Driven Threat Elicitation (EDTE)

The EDTE model is working as a sieve to sift the inappropriate threats. With this approach, we start with a laundry list of common threats [13] grouped by network, host, and application categories. Next, apply the threat list to the given application architecture and screen out the threats matching its own web application category. Then, further filtering can be done to the result threat set according to the security requirements of the given web application. We will be able to rule some threats out because they do not apply to the scenario of the given application. As a result, a set of filtered threats specific to the given web application can be obtained.

Threat Classification. This section we use web application classification to filter the threats proposed in [13] according to the given security requirements. In order to illustrate the detailed the steps of our approach, threats list used as the threats set to be filtered is described below.

For comprehensiveness, we choose the threat set proposed by [13] which enumerates the top threats that affect web applications at the network, host, and application levels. For the sake of being used conveniently in our approach, two related factors, web application category and CIA requirement are added in Table 4, Table 5 and Table 6.

Table 4. Network level threat [13]

No.	Threat Name and Description	WA Category	CIA Risk
1	**Information Gathering** Information (Network device type, operating system and application versions) may be detected by port scanning in order to perform attack	WA3 WA4 WA5 WA6	C
2	**Sniffing** Monitoring traffic on the network for data such as plaintext passwords or configuration information	WA3 WA4 WA5 WA6	C
4	**Spoofing** Spoofing may be used to hide the original source of an attack or to work around network access control lists (ACLs) that are in place to limit host access based on source address rules	WA3 WA4 WA5 WA6	C I A
5	**Session Hijacking** Session hijacking deceives a server or a client into accepting the upstream host as the actual legitimate host. Instead the upstream host is an attacker's host that is manipulating the network so the attacker's host appears to be the desired destination	WA3 WA4 WA5 WA6	C I A
6	**Denial of Service** Denial of service denies legitimate users access to a server or services. The SYN flood attack is a common example of a network level denial of service attack	WA3 WA4 WA5 WA6	A

Table 5. Host level threat [13]

No.	Threat Name and Description	WA Cate-gory	CIA Risk
7	**Viruses, Trojan horses, and Worms** Although these three threats are actually attacks, together they pose a significant threat to web applications, the hosts these applications live on, and the network used to deliver these applications	ALL	C I A
8	**Footprinting** Examples of Footprinting are port scans, ping sweeps, and NetBIOS enumeration that can be used by attackers to glean valuable system-level information to help prepare for more significant attacks	WA4 WA6	C
9	**Password Cracking** The attacker cracks the password if the default account names are used. The use of blank or weak passwords makes the attacker's job even easier	ALL	C I
10	**Denial of Service** An attacker can disrupt service by brute force against your application, or an attacker may know of a vulnerability that exists in the service your application is hosted in or in the operating system that runs your server	ALL	A
11	**Arbitrary Code Execution** If an attacker can execute malicious code on your server, the attacker can either compromise server resources or mount further attacks against downstream systems	WA3 WA4 WA5 WA6	C I A
12	**Unauthorized Access** Inadequate access controls could allow an unauthorized user to access restricted information or perform restricted operations	ALL	C I A

Table 6. Application level threat by application vulnerability category [13]

Input Validation

13	**Buffer Overflow** Buffer Overflow exploits are attacks that alter the flow of an application by overwriting parts of memory	WA3 WA4 WA5 WA6	C I A
14	**Cross-Site Scripting (XSS)** An XSS attack can cause arbitrary code to run in a user's browser while the browser is connected to a trusted Web site	WA4 WA6	C I A
15	**SQL Injection** A SQL injection attack exploits vulnerabilities in input validation to run arbitrary commands in the database	WA3 WA4 WA5 WA6	C I A
16	**Canonicalization** Canonicalization attacks can occur anytime validation is performed on a different form of the input than that which is used for later processing.	WA4 WA6	C I A
Authentication			
17	**Network Eavesdropping** An attacker armed with rudimentary network monitoring software on a host on the same network can capture traffic and obtain user names and passwords	WA3 WA4 WA5 WA6	C
18	**Brute Force Attacks** Brute force attacks rely on computational power to crack hashed passwords or other secrets secured with hashing and encryption	WA4 WA6	C I A
19	**Dictionary Attacks** An attacker uses a program to iterate through all of the words in a dictionary (or multiple dictionaries in different languages) and computes the hash for each word	WA4 WA6	C I A
20	**Cookie Replay** An attacker captures the user's authentication cookie using monitoring software and replays it to the application to gain access under a false identity	WA3 WA4 WA5 WA6	C I A
21	**Credential Theft** Credential theft occurs when an attacker obtains and uses valid account credentials (username and password) for unauthorized access to a computer	ALL	C I A
Authorization			
22	**Elevation of Privilege** An attacker may try to elevate privileges to a powerful account such as a member of the local administrators group or the local system account	ALL	C I A
23	**Disclosure of Confidential Data** The disclosure of confidential data can occur if sensitive data can be viewed by unauthorized users	ALL	C
24	**Data Tampering** Data tampering refers to the unauthorized modification of data	ALL	I A
25	**Luring Attacks** A luring attack occurs when an entity with few privileges is able to have an entity with more privileges perform an action on its behalf	WA3 WA4 WA5 WA6	C I A
Configuration Management			
26	**Unauthorized Access to Administration Interfaces** Malicious users able to access a configuration management function can potentially deface the Web site, access downstream systems and database	WA4 WA6	C I A
27	**Unauthorized Access to Configuration Stores** Because of the sensitive nature of the data maintained in configuration stores, you should ensure that the stores are adequately secured	WA4 WA6	C I A
28	**Retrieval of Clear Text Configuration Data** Sensitive data such as passwords and connection strings should be encrypt in that it helps prevent external attackers from obtaining sensitive configuration data	WA3 WA4 WA5 WA6	C
29	**Lack of Individual Accountability**	WA3 WA4	C

	Lack of auditing and logging of changes made to configuration information threatens the ability to identify when changes were made and who made those change	WA5 WA6	I A
30	**Over-Privileged Process and Service Accounts** If application and service accounts are granted access to change configuration information on the system, they may be manipulated to do so by an attacker	WA4 WA6	C I A
Sensitive Data			
31	**Access sensitive data in storage** Sensitive data must be secured in storage to prevent malicious users from gaining access to and reading the data	ALL	C I
32	**Network Eavesdropping** An attacker uses network monitoring software to capture and potentially modify sensitive data	WA4 WA6	C
33	**Data Tampering** Data tampering refers to the unauthorized modification of data, often as it is passed over the network	ALL	I
Session Management			
34	**Session Hijacking** A session hijacking attack occurs when an attacker uses network monitoring software to capture the authentication token (often a cookie) used to represent a user's session with an application	WA4 WA6	C
35	**Session Replay** Session replay occurs when a user's session token is intercepted and submitted by an attacker to bypass the authentication mechanism	WA4 WA6	C I A
36	**Man in the Middle** A man in the middle attack occurs when the attacker intercepts messages sent between you and your intended recipient	ALL	C
Cryptography			
37	**Poor Key Generation or Key Management** Attackers can decrypt encrypted data if they have access to the encryption key or can derive the encryption key	WA3 WA4 WA5 WA6	C
38	**Weak or Custom Encryption** Weak encryption algorithm provide no security if the encryption is cracked or is vulnerable to brute force cracking. Custom algorithms are particularly vulnerable if they have not been tested	WA3 WA4 WA5 WA6	C
39	**Checksum Spoofing** Some Hash algorithm can be interpreted and changed	WA3 WA4 WA5 WA6	C I
Parameter Manipulation			
40	**Query String Manipulation** The application is vulnerable to attack if the query string values represent sensitive data such as monetary amounts	WA3 WA4 WA5 WA6	C A
41	**Form Field Manipulation** Form fields of any type can be easily modified and client-side validation routines bypassed	WA4 WA6	C A
42	**Cookie Manipulation** Cookie manipulation is the attack that refers to the modification of a cookie, usually to gain unauthorized access to a Web site	WA4 WA6	C I
43	**HTTP Header Manipulation** An attacker may have to write his own program to perform the HTTP request, or he may use one of several freely available proxies that allow easy modification of any data sent from the browser	WA3 WA4 WA5 WA6	I A
Exception Management			
44	**Attacker Reveals Implementation Details** Internal implementation details such as exception details should not being reviewed by an attack which can greatly help them exploit	WA3 WA4 WA5	C

	potential vulnerabilities and plan further attack	WA6	
45	**Denial of Service** Attackers will probe a web application, usually by passing deliberately malformed input	WA3 WA4 WA5 WA6	A
	Auditing and Logging		
46	**User Denies Performing an Operation** The issue of repudiation is concerned with a user denying that he or she performed an action or initiated a transaction	ALL	C I
47	**Attacker Exploits an Application Without Trace** System and application-level auditing is required to ensure that suspicious activity does not go undetected	ALL	C I
48	**Attacker Covers His or Her Tracks** Your log files must be well-protected to ensure that attackers are not able to cover their tracks	ALL	C I

Algorithm. In this section, the process of filtering threats from common threat list according to its web application type and security requirements is described in the following algorithm. Starting from the web application classification, each threat in common threat list is sieved by the rule, as a result, a threat list applying for the given web application can be obtained.

Just like described in [13], "a threat is any potential occurrence, malicious or otherwise, that could harm an asset. In other words, a threat is any bad thing that can happen to your assets". It is meaningless to discuss threats without connection to their assets. Hence, it is necessary to associate the threats to their comprised assets so that web application developer can design proper security mechanism to protect the assets.

Algorithm 1

Step 1: Classification
Classify the given web application into one of the proposed web application type according to three attributes, use WAi to represent

Step 2: Rating the security requirements CIA of WAi
Web application is rated "Low", "Medium", or "High" on the metrics of Integrity, Availability, and Confidentiality, use {CIA requirements} to represent

Step 3: Filtering
for *all threats* T_i in *common threat list CTL* **do**
if T_i. *WA Category == All* **then**
$T_i \rightarrow \{TL\}$ /*Insert T_i to Threat List TL*/
end if
if *WAi. WA Category* $\in T_i$. *WA type* **then**
$T_i \rightarrow \{TL\}$ /*Insert T_i to TL*/
end if
end for

Step 4: Further Filtering
for *all* TL_i in *TL* **do**
if TL_i. *CIA risk does not match the {CIA requirements}* **then**
$TL_i \leftarrow \{TL\}$ /*Remove TL_i from TL*/
end if
end for

Notes: WA_i: the ith type of web application classification CTL: Common threat list
T_i: the ith threat type TL: threat list of the WAi
C:Confidentiality I: Integrity A: Availability

Threat Risk Analysis. A threat list can be obtained by using our EDTE method. Threats should be quantified in order to perform a comprehensive risk assessment for the target system. In this case, we use DREAD [36] to rate the security risk for each threat. DREAD is part of a system for classifying computer security threats used at Microsoft. DREAD is a classification scheme for quantifying, comparing and prioritizing the amount of risk presented by each evaluated threat. The DREAD acronym is formed from the first letter of each category below. DREAD stands for:

– Damage Potential: defines the amount of potential damage that an attack may cause if successfully executed.

- Reproducibility: defines the ease in which the attack can be executed and repeated.
- Exploitability: defines the skill level and resources required to successfully execute an attack.
- Affected Users: defines the number of valid user entities affected if the attack is successfully executed.
- Discoverability: defines how quickly and easily an occurrence of an attack can be identified.
- The calculation always produces a number between 0 and 10, the higher the number, the more serious the risk. Here is use case [36] of how to quantify the DREAD categories in this paper.

Table 7. DREAD Use Cases

Category	Use cases	Value
Damage potential	Leaking trival information	0
	Individual user data is compromised or affected	5
	Complete system or data destruction	10
Reproducibility	Very difficult to reproduce	0
	One or two steps required	5
	Just a web browser, without authentication	10
Exploitability	Very skilled	0
	Malware or attack tool available	5
	Novice programmer	10
Affected users	None	0
	Some users	5
	All users	10
Discoverability	Unlikely	0
	Accessible only to few users	5
	Published	10

3.4 Vulnerability Analysis

Once the credible threats are identified, a vulnerability analysis must be performed. The vulnerability analysis considers how to identify vulnerabilities and their potential impact of loss from a successful attack as well as the vulnerability of the facility/location to an attack.

Existing vulnerability identification tools for application can be used to perform vulnerability analysis. Vulnerability analysis is performed for the application assets, but not for data asset. Furthermore, vulnerabilities of applications used in server should be analyzed as well as the database program.

There are a number of vulnerability "scoring" systems such as Common Vulnerability Scoring System (CVSS). We can perform CVSS to rate each vulnerability identified from one of the diagnosis tools, and produce a total score for the vulnerability of system.

4 Case Study

In order to illustrate the usefulness of web application classification this paper will examine a case study used in [4]. Widgets Incorporated is a medium-sized consumer goods company. They have determined the need to create I-Tracker: a custom-built inventory tracking application to facilitate growing customer demand. The most common use case will be for sales staff to enter data from a sales order which will automatically update the inventory levels and alert the logistics staff to prepare the order for shipment. When the inventory level for a particular widget drops below a certain threshold the manufacturing division will be notified. The main types of data used in the application include inventory levels, customer IDs, sales orders numbers, descriptions of orders, and product IDs.

I-Tracker will be used by 30 internal users spread across the manufacturing, sales, and logistics departments, and that number is anticipated to grow to as much as 100 in the next few years. The business has indicated that the application may need to interface with a partner Widget Accessory supplier in the future. Widgets Incorporated currently receive 50-60 orders per day and anticipates that number grow to around 150. Data flow diagram of I-Tracker is shown in Fig. 5.

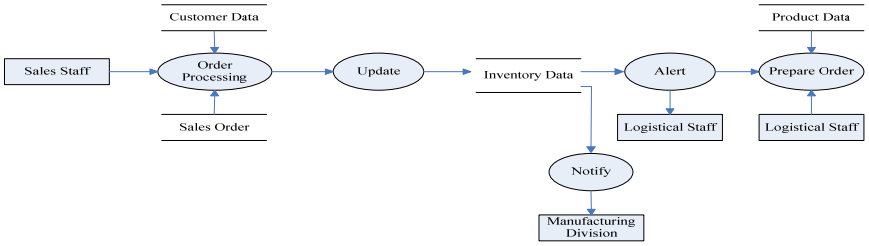

Fig. 5. Data flow diagram of the I-Tracker

Phase1. Architecture and Environment Analysis

From the case description, the example web application is used within internal environment facing a number of known users. Thus, it belongs to the first type of web application categories.

Phase2. Asset analysis

Based on the description of section 3, assets can be identified as data store, data flow and processes. Process is the dynamic execution of application, while data flow is the dynamic representation of a "flow" of data in the system.

Table 8. Asset criticality analysis

ID	Asset Name	Type	Personal or sensitive	Environment	Criticality	Rating value
1	Customer data	Data	Yes	Internal use	Medium	5
2	Sales order	Data	Yes	Internal use	Medium	5
3	Inventry data	Data	No	Internal use	Very Low	1
4	Product data	Data	No	Internal use	Very Low	1
5	Order processing	Application	Yes	Internal use	Medium	5
6	Order update	Application	Yes	Internal use	Medium	5
7	Alert logistics staff	Application	No	Internal use	Very Low	1
8	Inventry level notify	Application	No	Internal use	Very Low	1
9	Prepare data	Application	No	Internal use	Very Low	1

Table 9. Threat list after filtering

ID	Threat Name	CIA
7	Viruses, Trojan horses, and Worms	CIA
9	Password Cracking	CI
10	Denial of Service	A
12	Unauthorized Access	CIA
21	Credential Theft	CIA
22	Elevation of Privilege	CIA
23	Disclosure of Confidential Data	C
24	Data Tampering	CI
31	Access sensitive data in storage	C
36	Man in the Middle	C
46	User Denies Performing an Operation	CI
47	Attacker Exploits an Application Without Trace	CI
48	Attacker Covers His or Her Tracks	CI

Phase3. Threat analysis

Step1: Threat identification

1. Security requirement rating. It is necessary to rate the security requirements CIA of target application when using our EDTE method. Using internal guidelines based on documents such as [21], the following application classification may be produced:

– Confidentiality: Low

All data in the application is readily available to anyone in the company. Sensitive financial data and client private information are not handled by this application.

– Integrity: High

Poor inventory and shipping tracking may result in significant financial loss to the company and may result in customer dissatisfaction / loss of customers.

– Availability: Medium

A major disruption of the application will cause a backlog in shipping and have some financial consequences to the organization. Minor disruptions, however, can be tolerated as customers expect a 4-6 week delay in receiving their goods.

2. Filtering. According to the algorithm 1 described in Section 3.3, the most likely threats are filtered and listed in Table 9.

We can infer from the table above that internal attackers are the major factors to perform the attack.

3. Further Filtering. The threat list can be further screened out according to the security requirements of CIA aspects. In terms of the algorithm described in Section 3, threats with only Confidentiality (C) requirements can be rule out in that the given application has low requirements on Confidentiality, while threats with Integrity (I) and Availability (A) are remained.

Step2: Threat risk quantification. DREAD is used to quantify the security level for each threat identified from our EDTE model according to the rating value in Table 7.

Table 10. Threat risk quantification

ID	Threat Name	D	R	E	A	D	Total
7	Viruses, Trojan horses, and Worms	10	10	10	10	10	10
9	Password Cracking	10	5	5	5	5	7
10	Denial of Service	10	5	5	10	0	6
12	Unauthorized Access	10	5	5	5	0	5
21	Credential Theft	10	0	5	5	5	5
22	Elevation of Privilege	10	10	0	5	0	5
24	Data Tampering	10	5	5	5	5	6
46	User Denies Performing an Operation	5	5	5	5	5	5
47	Attacker Exploits an Application Without Trace	5	5	5	0	0	3
48	Attacker Covers His or Her Tracks	5	5	5	0	0	3

Phase4: Vulnerability analysis

In this paper we do not give a detailed vulnerability identification method. However, some common vulnerabilities in design phase can be included due to their prevalence. Suppose we use one of the host vulnerability tools to identify the vulnerabilities and an overall evaluation number between 1 and 10 can be obtained by using CVSS.

Phase5: Potential risk

From previous steps, the average score for asset and threat is 2.78 and 5 respectively. We just suppose the vulnerability score of target application is 3.

According to formula (2), security vector of the target application $SV=\sqrt{(A^2+T^2+V^2)/3}=\sqrt{(2.78^2+5^2+3^2)/3}\approx 3.74$. From Table 11, we can conclude that the security risk of target web application is low.

Table 11. Risk Rating Scale

Value	Treat severity	Rank
0.0-1.99	Very low	1
2.0-3.99	Low	2
4.0-5.99	Medium	3
6.0-7.99	High	4
8.0-10.0	Very high	5

5 Conclusions

Risk analysis is, at best, a good general-purpose yardstick by which we can judge our security design's effectiveness. Because roughly 50 percent of security problems are the result of design flaws, performing a risk analysis at the design level is an important part of a solid software security program. Taking the trouble to apply risk analysis methods at the design level for any application often yields valuable, business-relevant results [28].

This paper proposes a novel approach to perform risk assessment at design stage for web application which is based on multiple security vectors of asset, threat and vulnerability. The proposed web application classification can ease the elicitation of threats for the given application with the aid of web application classification which is defined taking consideration of the complexity and environments where the web applications are hosted. With the proposed method and result obtained under this work, it is possible to determine the appropriate design to help identify the most critical threats in the web application.

One defect of our proposed model is that it is incapable of identify the emerging threats out of the common threat list. Fortunately, the common threat list can be extended along with the emergency of the new threats. Another defect is that the filtered threats by the proposed model are not as specific as elicited by other methods. At last, we do not give the implementation of vulnerability identification. However, our proposed method can shorten the asset and threat elicitation time significantly and an experienced developer can easily relate the filtered threats to its appropriate scenery. Future research will focus on define or improve vulnerability identification approach in design phase and then improve the proposed approach to an automatic implementation tool for web application.

6 Acknowledgment

This work was sponsored by Liaoning Province Office of Education of China Project Research on Security-oriented Software Reengineering (Grant No. L2010439) and partial financial support by the National Natural Science Foundation of China (Grant No. 60873064 and Grant No. 90818026).

References

[1] B. D. R. Marino, H. M. Haddad, J. E. Molero A, "A Methodological Tool for Asset Identification in Web Applications," in *Proceeding of the 4th International Conference on Software Engineering Advances*, Porto, Portugal, pp. 413-418, 2009.

[2] Web Application Security Trends Report, http://www.cenzic.com/downloads/Cenzic_AppSecTrends_Q3-Q4-2008.pdf

[3] The Importance of Application Classification in Secure Application Development, http://www.webappsec.org/projects/articles/041607.shtml

[4] J. R. Maguire and H. G. Miller, "Web-application Security: From Reactive to Proactive," *IT Professional*, Vol. 12, No. 4, pp. 7-9, 2010.

[5] G. Sindre and A. L. Opdahl, "Eliciting Security Requirements with Misuse Cases," *Requirements Engineering*, Vol. 10, No. 1, pp. 34-44, 2005.

[6] I. F. Alexander, "Misuse Cases: Use Cases with Hostile Intent," *IEEE Software*, Vol. 20, No. 1, pp. 58-66, 2003.

[7] D. G. Firesmith, "Analyzing the Security Significance of System Requirements," in *Proceedings of Symposium on Requirements Engineering for Information Security*, Paris, France, 2005.

[8] F. A. Braz, E. B. Fernandez, M. VanHilst, "Eliciting Security Requirements through Misuse Activities," in *Proceeding of the 19th International Workshop on Database and Expert Systems Applications*, Turin, Italy, pp. 328-333, 2008.

[9] F. Swiderski and W. Snyder, Threat modeling, Microsoft Press, Redmond, Washington, 2004.

[10] I. A. Tondel, M. G. Jaatun, P. H. Meland, "Security Requirements for the Rest of Us: A Survey," *IEEE Software*, Vol.25, No. 1, pp. 20-27, 2008.

[11] S. Myagmar, A.J. Lee, W. Yurcik, "Threat Modeling as a Basis for Security Requirements," in *Proceeding of Symposium on Requirements Engineering for Information Security*, Paris, France, 2005.

[12] M. A. Hadavi, H. Shirazi, H. M. Sangchi, V. S. Hamishagi, "Software Security: A Vulnerability-activity Revisit," in *Proceeding of the 3rd International Conference on Availability, Reliability and Security*, Barcelona, Spain, pp. 866-872, 2008.

[13] J.D. Meier, A. Mackman, S. Vasireddy, M. Dunner, R. Escamilla, A. Murukan, Improving Web Application Security: Threats and Countermeasures, Microsoft Press, Redmond, Washington, 2003.

[14] C. Möckel and A. E. Abdallah, "Threat Modeling Approaches and Tools for Securing Architectural Designs of an E-banking Application," in *Proceeding of the 6th International Conference on Information Assurance and Security*, Atlanta, GA, USA, pp. 149-154, 2010.

[15] An Approach to Web Application Threat Modeling, http://www.infosecwriters.com/text_resources/pdf/AShrivastava_Web_Application_Threat_Modeling.pdf

[16] E. A. Oladimeji, S. Supakkul, L. Chung, "Security Threat Modeling and Analysis: A Goal-oriented Approach," in *Proceedings of the 10th International Conference on Software Engineering and Applications*, Dallas, Texas, USA, 2006.

[17] M. Jackson, "Problem Frames and Software Engineering," *Expert Systems*, Vol. 25, No. 1, pp. 7-8, 2008.

[18] C. B. Haley, R. C. Laney, J. D. Moffett, B. Nuseibeh, "Security Requirements Engineering: A Framework for Representation and Analysis," *IEEE Transaction On Software Engineering*, Vol. 34, No. 1, pp. 133–153, 2008.

[19] D. Hatebur, M. Heisel, H. Schmidt, "Analysis and Component-based Realization of Security Requirements," in *Proceedings of the 3rd International Conference on Availability, Reliability and Security*, Barcelona, Spain, pp. 195-203, 2008.

[20] J. P. Jesan, "Threat Modeling Web-applications Using STRIDE Average Model," in *Proceedings of Computer Security Conference*, Myrtle Beach, USA, 2008.

[21] Protecting Sensitive Compartmented Information within Information Systems, http://www.fas.org/irp/offdocs/DCID_6-3_20Manual.htm

[22] L. Liu, E. S. K. Yu, J. Mylopoulos, "Secure-i*: Engineering Secure Software Systems through Social Analysis," *International Journal of Software and Informatics*, Vol. 3, No. 1, pp. 89-120, 2009.

[23] L. Liu, E. Yu, J. Mylopoulos, "Secure Design Based on Social Modeling," in *Proceedings of the 30th Annual International Computer Software and Applications Conference*, Chicago, IL, USA, pp. 71-78, 2006.

[24] T. Long, L. Liu, Y.J. Yu, Z. Jin, "AVT Vector: A Quantitative Security Requirements Evaluation Approach based on Assets, Vulnerabilities and Trustworthiness of Environment," in *Proceedings of the 17th IEEE International Requirements Engineering Conference*, Atlanta, Georgia, USA, pp. 377-378, 2009.

[25] H. Guan, W.R. Chen, L. Liu, H.J. Yang, "Environment-driven Threats Elicitation for Web Application", in *Proceeding of Agent and Multi-Agent Systems: Technologies and Applications*, Manchester, UK, Vol. 6682 , pp. 291-300, 2011.

[26] BSI, Code of Practice for Information Security Management, British Standards Institute, London, 1999.

[27] Y.J. Chung, I.J. Kim, N.H. Lee, T. Lee, H. P. In, "Security Risk Vector for Quantitative Asset Assessment," in *Proceeding of International Conference on Computational Science and Its Applications*, Singapore, Vol. 3481, pp. 274-283, 2005.

[28] D. Verdon and G. McGraw, "Risk Analysis in Software Design," *IEEE Security and Privacy*, Vol. 2, No. 4, pp.79-84, 2004.

[29] I. Mkpong-Ruffin, D. A. Umphress, J. Hamilton, J.Gilbert, "Quantitative Software Security Risk Assessment Model," in *Proceeding of the 3rd ACM Workshop on Quality of Protection*, Alexandria, VA, USA, pp.31-33, 2007.

[30] Risk Management Guide for Information Technology Systems, http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf

[31] Methodology for Information Systems Risk Analysis and Management, https://www.ccn-cert.cni.es/publico/herramientas/pilar43/en/magerit/meth-en-v11.pdf

[32] A Complete Guide to the Common Vulnerability Scoring System Version 2.0, http://www.first.org/cvss/cvss-guide.html#i2.2.1

[33] OCTAVE, http://www.cert.org/octave/

[34] D. D. Cock, K. Wouters, D. Schellekens, D. Singelee, B. Preneel, "Threat Modelling for Security Tokens in Web Applications," in *Proceeding of the 8th Conference on Communication and Multimedia Security*, Windermere, UK, pp. 213-223, 2004.

[35] L. Jiang, H. Chen, F. Deng, "A Security Evaluation Method Based on STRIDE Model for Web Service," in *Proceeding of the 2nd International Workshop on Intelligent Systems and Applications*, Wuhan, China, pp. 1-5, 2010.

[36] Threat Risk Modeling, https://www.owasp.org/index.php/Threat_Risk_Modeling